JPL D-16624

Mission Data System

Software Management Plan

Version 1.0

August 16, 1999

Prepared by:

Anne Elson, MDS Process Engineer	Date

Approved by:

Al Sacks, MDS Project Manager	Date

Jet Propulsion Laboratory
California Institute of Technology

Change Log

Change Log – 1/25/99 (draft 3)

1/25/99 – Move application of Bob Rasmussen's control architectural patterns to architectural design section

Change Log – 2/3/99 (draft 4)

1/28/99 – Add up front discussion how resource constraints may require customization, subsetting of processes described in this and the other MDS process documents
2/1/99 – ISO 9000-3 software compliance matrix started
2/2/99 – Respond to, add updates based on Bob Barry's comments

Change Log – 2/16/99 (draft 5)

2/3/99 – Continue to update based on Bob Barry's comments
2/3/99 – Include Susan Roberts comments, add D number
2/3/99 – Change DS-4 to ST-4
2/4/99 – Put matrix in that shows our compliance to SPDP shall statements and ISO, remove other ISO compliance matrix (redundant)
213/99 – Clean up lifecycle phases so that they follow viewgraph presentation for peer review
8/7/99 – More clean up prior to sign off of preliminary version

Change Log – 8/16/99

8/10/99 – Release Preliminary Version for review
8/16/99 – Release Version 1.0

1 INTRODUCTION

1.1 Identification

This document is the software management plan for the Mission Data System Development Project. It describes the MDS software development lifecycle. The software development process documented herein applies to all software being produced within the MDS project.

The MDS software development process and the team infrastructure supporting it are currently under development. As the MDS team evolves its processes and infrastructure this document and other related MDS process documents will be updated to reflect those changes. Readers should look in the MDS Project Library for the latest electronic version of this document.

MDS process documents describe a complete software development lifecycle process. Current MDS budget, schedule and workforce constraints may prevent immediate full implementation of this process. MDS process documents will provide tailoring guidelines so MDS teams can streamline processes and products to fit within their current resource and schedule constraints.

1.2 Project Definition Overview

1.2.1 Product Description

The Mission Data System Development Project (MDS) is tasked with building unified flight, ground and test software for future JPL space missions. The MDS team will develop a core set of standard flight, ground and test software capabilities for JPL space mission customers. In addition the MDS team will provide a set of partially fleshed out software capabilities (frameworks) and example implementations for capabilities that are typically mission unique or have mission unique aspects. Customer missions may use or adapt MDS reference examples and/or they may fill in and build upon the relevant frameworks to meet their mission specific needs.

1.2.2 MDS Software Development Objectives

MDS project is charged with developing a unified flight/ground/test system for JPL space missions. The MDS project is working to align and consolidate resources, technology and organizations for the achievement of this purpose. The MDS project will develop an end-to-end mission data system and an implementation approach for its development that eliminates redundancy. This implementation approach will result in new development and test processes and environments. The MDS project will develop a mission data system that is simple and cost effective both to implement and to operate. This vision includes the concept of a flight system that can operate autonomously with little or no intervention from the ground. MDS software will be designed to be easily adapted, extended and/or customized by JPL mission customers with a variety of hardware platforms and mission objectives. Additionally the MDS project will be working to develop end-to-end mission data system capabilities that enable new mission technologies and science opportunities. These objectives are in alignment with the lab concept of "Better, Faster, Cheaper" missions. The MDS team will be working to balance the specific needs of current customer missions against the need for the MDS project to move ahead with longer range project objectives.

1.2.3 Customer Definition and Advocacy

The X2000/MDS 1st Delivery Project (FDP) is the first MDS customer. The MDS team will produce test software to verify X2000 avionics hardware. Additionally the MDS team will produce flight, ground and test software for a reference mission that will be tested on X2000 avionics hardware. This fictitious mission will be designed to demonstrate MDS architectural features, to act as an adaptation example for MDS mission customers and to show MDS compatibility with the mission software needs of the first MDS mission customers. The X2000/MDS 1st Delivery Project will build a test bed (PT-MDS) upon which MDS reference mission software will exercised. The X2000/MDS 1st Delivery integration and test team will validate MDS reference mission software using a set of reference mission scenarios developed by the MDS team. These scenarios will be based on requirement inputs and mission scenarios from the first MDS customer missions. Additionally these scenarios will include activities to demonstrate some MDS architectural features and capabilities independent of first mission customer needs.

The Outer Planets/Solar Probe (OP/SP) project is the first MDS mission customer. The OP/SP Project has three missions with launches in 2003, 2004 and 2007. The first two OP/SP missions, Eruopa Orbiter and Pluto Express, will use X2000 avionics and MDS software. The third OP/SP mission, Solar Probe, will be built and operated by an industry partner. This partner will have the option of using X2000 avionics hardware and/or MDS software or providing hardware and/or software from another source. Several OP/SP team members are working closely with the MDS and the X2000/MDS 1st Delivery teams to ensure that the technical and programmatic decisions made by the MDS and the X2000/MDS 1st Delivery teams are in alignment with the mission needs of the OP/SP Project. Where feasible OP/SP personnel will join the MDS and/or X2000/MDS 1st Delivery teams to help with the development and testing of X2000 hardware and MDS software. MDS team plans to continue this pattern of customer participation throughout the MDS project life cycle. The OP/SP Project sees their participation as a way of familiarizing and training project personnel before they assume development and/or operations duties within the OP/SP project. MDS will offer the same type of participatory relationship to other customer missions as they come on-board.

MDS is negotiating to supply software to several other JPL deep space missions that are currently in conception or under development.

1.2.4 Project Performance Goals and Success Criteria

MDS must satisfy the mission software needs of both current and future customers. Thus the MDS development approach is to balance the immediate flight, ground, test and operations software needs of its current customer missions with the need to evolve these capabilities for future missions. MDS products must be flexible enough to evolve gracefully. Future innovation will be required to enable future missions. MDS will not be successful until current customer missions are successful and MDS will not truly be successful if only the current customer missions are successful.

MDS performance monitoring is part of its software risk management process. See the metrics and risk management sections in MDS Software Quality Assurance Plan.

1.2.5 Project Timeframe

The first phase of the MDS project covers the development and delivery of MDS software to the X2000/MDS 1st Delivery Project via a series of software deliveries on 6 month centers. Each delivery will include increasing capabilities and will culminate in a final delivery that will form the core of the flight and ground software needed to launch and to fly both the Europa Orbiter and the Pluto Express Spacecraft in quiescent cruise. The X2000/MDS 1st Delivery Project will integrate and test MDS software in the X2000 provided test bed designed for this purpose prior to its formal release to the OP/SP project. The OP/SP project may take informal deliveries of MDS software directly from the MDS team prior to its official release.

This first phase of MDS development will be followed by a second phase of development during which MDS will make a series of deliveries directly to its OP/SP mission customer. These deliveries will consist both of negotiated updates (if deemed necessary by the OP/SP project) to the original delivered MDS software and planned enhancements to MDS/TMOD jointly developed ground and flight based navigation software. The navigation software enhancements are needed by both the Europa Orbiter and Pluto Express missions in time for spacecraft arrival at the Jovian system as well as during subsequent portions of these missions. The current schedule for the navigation software updates is a set of once yearly releases from 2002 through 2006 when the last planned navigational updates are planned to be complete.

MDS will also support X2000 2nd delivery project (if this project materializes) and X2000 2nd delivery mission customers. Customers and details are TBD.

1.2.6 MDS Software Classification

MDS will develop Class A, Class B and Class C software. See discussion of software classifications in Section 1 in JPL D-15378, "The JPL Software Development Process Description".

1.3 Document Overview

This document is the MDS Software Management Plan (MDS SMP). Its organization follows the template for a software development management plan provided in Appendix A of JPL D-15378, "The JPL Software Development Process Description (SDPD)". This process description both levies requirements on and establishes good software development practices. It also responds to the ISO 9000-3 guidelines. The MDS SMP responds to the requirements levied on software development processes by the SDPD. Any departure of the MDS SMP from requirements levied by the SDPD will be documented herein. The MDS SMP is in compliance with the MDS Project Implementation Plan (MDS PIP) and responds to controlling JPL process and policy documents cited below and/or listed in the MDS PIP.

1.4 Referenced Documents

Latest versions of JPL process and policy documents will be found at the JPL DMIE web site http://dmie. Latest versions of MDS documents will be found in the MDS Project Library.

The JPL Software Development Process Description, JPL D-15378, Revision C, Version 3.0, October 5, 1998

MDS Project Implementation Plan, JPL D-16623, Draft, January, 1999

Quality Assurance Master Plan, JPL-D16000, November, 1998

Risk Management Handbook for JPL Projects, JPL D-15951, October, 1998

1.5 Notation

Use of "shall" shall denote a process or product requirement on project personnel and shall be met by them unless explicitly waived by the project manager.

2 PROJECT ORGANIZATION

MDS project organization is described in detail in the MDS PIP. Where descriptions of the MDS organization are provided in this document they are provided only as an aid to the reader. The MDS PIP is the controlling document.

3 WORK BREAKDOWN STRUCTURE, PROJECT RESOURCES AND SCHEDULE

See the MDS PIP and MDS web site for details.

4 PROJECT SOFTWARE INPUTS FROM EXTERNAL SOURCES

4.1 Inputs from Customers

4.1.1 Customer Mission

MDS customers will participate with the MDS team in a partnering relationship throughout the MDS software lifecycle. The software lifecycle description below provides a general description this lifecycle relationship. At the start of an MDS customer relationship MDS will negotiate a memorandum of understanding (MOU) and a delivery plan with the customer mission. The MOU will be written first and will capture at a very high level the initial agreements between MDS and the customer. This will be followed by a delivery plan that will define in much more detail both the contents of the MDS product to be produced and its schedule for completion. Various programmatic agreements such as how resources are shared, what training will be provided by MDS, how tools will be shared and licensed, etc., will be captured in the delivery plan. An MDS customer mission will provide inputs to MDS to help in the writing of this plan. These inputs will provide MDS insight into customer mission goals and constraints. Inputs should include customer mission plan, any requirements that directly or indirectly reflect needed software capabilities and constraints, mission operations scenarios, spacecraft mechanical, electrical and structural designs, hardware software interface descriptions and any other technical material that will aid MDS and customer personnel in establishing a good plan.

4.1.2 X2000 Program

MDS will work together with X2000 projects (currently X2000/MDS 1st Delivery Project) to produce integrated, generic avionics subsystems for mission customers.

4.2 Inputs from separate JPL Organizations

4.2.1 MDS TMOD interface

The MDS project is part of TMOD. The MDS project is charged with aligning and consolidating resources, technology and organizations both internal to and external to TMOD for the efficient and cost effective development, test and operation of end-to-end mission data systems. The MDS project will establish new development processes, and development and test environments as part of its charter. MDS will work to replace, extend and/or evolve legacy systems as the project integrates MDS organization, practices and processes into existing organizational elements at the lab.

MDS will coordinate the development of cross-cutting TMOD software products with TMOD elements external to itself. MDS will deliver integrated MDS TMOD end-to-end data system products to MDS customers. MDS will help mission customers estimate TMOD service costs for external TMOD services likely to be impacted by the MDS development effort.

4.2.2 MDS DNP interface

MDS is working with DNP to define a new software development process for the lab. MDS is also working with DNP to make use of existing DNP processes, infrastructure and tools where appropriate within the MDS effort.

4.3 MDS management of customer and third party supplied inputs

MDS will validate externally produced software products prior to merging them with or using them on MDS internally produced software products. Validation approach will be documented in the MDS Validation Test Plan. Configuration management of customer and third party supplied software artifacts is described in the MDS Configuration Management Plan.

MDS will configuration manage non-software customer work products that are required customer inputs in support of MDS lifecycle activities (includes customer requirements, customer mission scenarios, customer spacecraft block diagram, etc.). Configuration management of customer supplied items will begin with the sign off of the MDS Customer Delivery Plan.

5 ASSUMPTIONS, CONSTRAINTS, AND RISKS

5.1 Assumptions regarding MDS customer interface

The MDS customer interface may take on one of several flavors. MDS and a customer project may work together as one team to develop MDS products for customer mission. This may include the collocation of customer software personnel within the MDS team for the duration of the joint MDS customer development effort. Alternatively a customer project may hire MDS personnel to do the entire software development effort including the complete adaptation, extension and/or customization of MDS product for their specific mission. Customers may adapt some combination of these two approaches. Details of a specific MDS customer relationship will be negotiated on a customer by customer basis and captured in the MDS customer delivery plan.

5.2 Software constraints

Software design constraints will be captured in various MDS design notes (including Application Program Interface specifications (APIs)) and in the MDS Architectural Design document (global design patterns). Software coding constraints, both language specific and language non-specific are captured in MDS coding standards document(s). The most current version of these documents will be maintained in the MDS Project Library (DocuShare).

MDS plans to make extensive use of COTs software. The project will use COTs instead of "home grown" software (both tools and application software) where possible and applicable. MDS plans to establish licensing and maintenance agreements that will include customers. Customers will get the benefit of the group procurement but will be expected to pay for their copies of the licenses. Some commercial companies may have licensing restrictions that will preclude inclusion of mission customer partners who are external to the lab in a single licensing agreement. Under these circumstances the customer mission project will be responsible for negotiating license arrangements independent of MDS.

MDS is charged with evolving and/or replacing legacy ground software systems gracefully. The MDS team will focus on developing the flight software portion of its end-to-end mission data system first. The team is planning to make use of legacy ground system software, particularly in the earlier phases of its 1st delivery development lifecycle. MDS will wrap this software to make it compatible with new interfaces and capabilities within the other parts of its evolving end-to-end mission data system.

Software security will be covered in the MDS Software Quality Assurance Plan. MDS shall meet JPL software security requirements.

MDS shall meet JPL software safety requirements as described in the MDS Software Quality Assurance Plan. The MDS Software Quality Assurance engineer will work with the rest of the MDS team to make sure that MDS is in compliance with these requirements.

5.3 MDS software risk management

MDS will follow the software risk management processes described in the risk management section of the MDS Software Quality Assurance Plan

6 SOFTWARE DEVELOPMENT PROCESS OVERVIEW

6.1 Software lifecycle in relation to MDS project lifecycle

MDS is a software development project. The MDS project lifecycle is a software development lifecycle although there are systems engineering activities in the outer loop portion of the lifecycle. See the sections on the MDS software lifecycle process below for details. This document does not discuss the MDS project formulation phase. This phase is described in the MDS PIP.

6.2 MDS Development Philosophies

6.2.1 Architecture as focal point of analysis and design

The MDS software development approach is to make architecture the focal point of the analysis and design effort. This approach proceeds from two central architectural principles and their corollaries:

Subsystems are constructed from architectural elements – not the other way around

- Find the problems in common
- Create common solutions
- Tailor the general solutions to the particular problems

Managing interactions is the foundation of a design

- Find the interaction mechanisms (decoupling where feasible)
- Otherwise, create coordination services for the interactions
- Control interactions through these common services rather than function-to-function

MDS addresses these principles by adopting a set of frameworks upon which the rest of the design is built. These frameworks are constructed around a few basic notions familiar to spacecraft system design. The most important of these is the notion of "State". State is defined as a representation of the momentary condition of an evolving system and is a central organizing theme of the MDS architecture. Models describe how a system's state evolves. State information and models together provide the user of a system with the information on how to operate that system, to determine or control its future, and to assess its performance. The MDS design effort proceeds by constructing subsystems from the architectural elements. This process is iterative and involves multiple passes though the MDS lifecycle, both the outer and the inner loops (see loop discussion below).

6.2.2 MDS project vision

MDS will always balance individual customer requests against the need to fulfill the long range MDS vision. In addition to producing software to meet a specific customer's end-to-end mission needs MDS will be working to define software development, software test and mission data system flight operations processes for itself and other future customers that will further the lab goal of achieving "Better, Faster, Cheaper" missions. MDS will also be working to develop end-to-end mission data systems that enable new mission technologies and science opportunities. MDS customers must be aware that the products that MDS develops for them will reflect trades between their short term customer needs and long term MDS objectives.

6.2.3 Object Oriented Analysis and Design Approach

MDS team will use object oriented analysis and design methodologies to implement their end-to-end mission data systems. The MDS team will be following Bruce Douglas' development process as described in his book "Doing Hardtime: Developing Real-Time Systems with UML Objects, Frameworks and Patterns". Details of Douglas' development approach will not be restated here. Readers should familiarize themselves with his process and terminology as references will be made to it throughout discussion of the MDS software lifecycle later in this document.

MDS is following an OOA/OOD development process because we believe this approach will:

1) increase the quality of the product;
2) improve the repeatability and predictability of the MDS development effort; and
3) decrease the amount of effort required of the team to produce quality products on time

The OOA/OOD development process that MDS follows makes use of a modeling language (Unified Modeling Language or UML) to develop different views (models) of the system under development. Bruce Douglass defines a system model as "an organized, internally-consistent set of abstractions that collaborate to achieve a system description at a desired level of detail and maturity." In each phase of the MDS software development life cycle the MDS team develops models to describe the evolving system. The team develops analysis models, design models, translation (source code) models and testing models. A very important concept at the root of this process is that all of the models developed by the team describe the same underlying system. The models all represent different views of that system and are not independent of one another. If the MDS team's understanding of the underlying system changes as they are developing a model of the system then that change needs to be reflected in the other model views of the system. The various models that the team uses to describe the underlying system should evolve in parallel (be kept consistent) as the team's understanding of the underlying system matures.

6.2.4 Gradual modification of legacy software systems

The MDS team will develop new end-to-end flight, ground and test mission software systems. As MDS develops new processes and the environments to host them legacy mission software systems will be impacted. Initially MDS will wrap legacy software to hide changes. Later it will sequence replacements to legacy systems so as to minimize the technical and programmatic impacts of the changes. An MDS goal is the gradual and graceful transformation of legacy systems into new systems that will meet the needs of future JPL deep space missions.

6.2.5 Continuous Integration

A key feature of the MDS software development approach is the concept of continuous integration. This concept requires that the evolving system be built up from objects and their interfaces, with the interfaces being defined and implemented first. This is in alignment with MDS architectural principles and a MDS development strategy that says that the team should design horizontally but implement vertically (this approach is also recommended by Douglass) .

MDS software products will start as skeletal systems. The earliest MDS builds will consist of software interface code and stubs for all of the major software components. By implementing interfaces first the team will produce an integrated system that can be exercised end-to-end very early in the delivery cycle. The team will then flesh out and/or enhance initial component capabilities to improve component fidelity to project requirements throughout the rest of the delivery cycle for a customer. This approach enables frequent and asynchronous integration of one or more pieces of the evolving MDS product without impact to other asynchronously evolving parts. Although top level MDS development schedules will show software release milestones on six month centers these should be viewed as points in time when the evolving MDS product achieves a certain level of capability. They should not be interpreted as signifying the end of an integration and test activity for a large, monolithic build effort. Between successive capability milestones there will be frequent, continuing informal integration of the evolving software.

6.2.6 Problem reporting and change requests

MDS, X2000 and initial MDS mission customers plan to use common and/or compatible tools and processes for tracking problem reports and change requests. Details will be provided in the MDS Software Quality Assurance Plan and in the MDS Configuration Management Plan.

6.3 Lifecycle Overview

The MDS software lifecycle can be viewed as a set nested loops: one outer loop and an associated set of inner loops. The MDS project traverses one cycle of its outer loop and multiple cycles of its inner loops for each formal delivery of MDS software to an MDS customer. A lifecycle model provides a set of development guidelines to the developers. It identifies a set of development phases and the work products (artifacts) to be produced in each. A lifecycle model helps to bring structure and standardization (predictability) to an activity that often appears to be chaotic and unpredictable. Douglass promotes a spiral development lifecycle model with iterative proto-typing. In this model the developers repeat a set of major lifecycle phases multiple times. During each iteration of the lifecycle the team, focusing on a particular set of capabilities, increases the overall capability of the evolving software system. Work products (artifacts) grow in their completeness and quality until all agreed upon system requirements and constraints are achieved (or re-negotiated). Although the MDS lifecycle can also be said to contain the concept of iterative proto-typing it is not a simple spiral lifecycle model. The MDS lifecycle approach with its outer and inner loop development efforts is designed to accommodate current JPL infrastructure, and a very complex development and maintenance environment. MDS is chartered with the development and maintenance of reusable, evolvable end-to-end mission data systems, the development of which may be tied both to a fixed launch schedule and to the parallel development of one or more state-of-the-art mission vehicles and their associated science and/or technology instruments.

MDS team may supply software to multiple customers in parallel. Under these circumstances the MDS team will be participating in more than one MDS software lifecycle activity at a time. Since a large portion of the MDS product should be common to all of its customers overlapping development cycles of concurrent customers should not be a large problem. MDS managers will need however to spend extra time and effort in both the planning and management of the overlapping activities to ensure successful completion of a single customer's lifecycle. In the MDS lifecycle the outer loop maps to MDS management and system engineering activities for an entire development and maintenance effort for a single MDS customer. The outer loop can be divided into 4 phases: feasibility, elaboration, construction and transition. An MDS outer loop cycle begins when MDS management contracts with a customer mission to produce that customer's core mission software. An outer loop cycle ends when the customer successfully adapts MDS core software products to their particular mission needs and MDS is no longer involved with either the development or maintenance of that software.

For each circuit of the outer loop MDS development teams complete multiple iterations (cycles) of a set of associated inner loops. Each MDS inner loop represents one software technical domain within the current MDS development effort. An inner loop is divided into 3 phases of software development: analysis and design, implementation, and evaluation and test. An MDS development team responsible for inner loop activities may traverse an inner loop multiple times during one cycle of the outer MDS loop. Inner loop cycles will run asynchronously to one another at least part of the time. During inner loop iterations however there will also be some planned, periodic alignments of inner loop completions across multiple software domains (sometimes referred to as synchronization points). These will occur when functionality crosses domains and requires coordination across teams and/or at 6 month intervals when a new copy of the evolving system will be provided to the MDS verification team and then to the X2000/MDS 1st delivery team for evaluation and test. Each iteration of an inner loop by one or more of the MDS software development teams will be concluded with an incremental, informal release of that portion of evolving MDS product. The updated MDS product will then be baselined and will be made available to customers for informal evaluation and test. When all inner loop cycles are complete within a single outer loop cycle and system level verification completed for the entire set of software MDS will formally deliver a baselined set of MDS software products to their mission customer. It should be noted this process is being modified for MDS delivery to its OP/SP mission customer. The final MDS software system will be delivered to X2000/MDS 1st Delivery Project for further test and integration on X2000 avionics hardware prior to formal release to the OP/SP project.

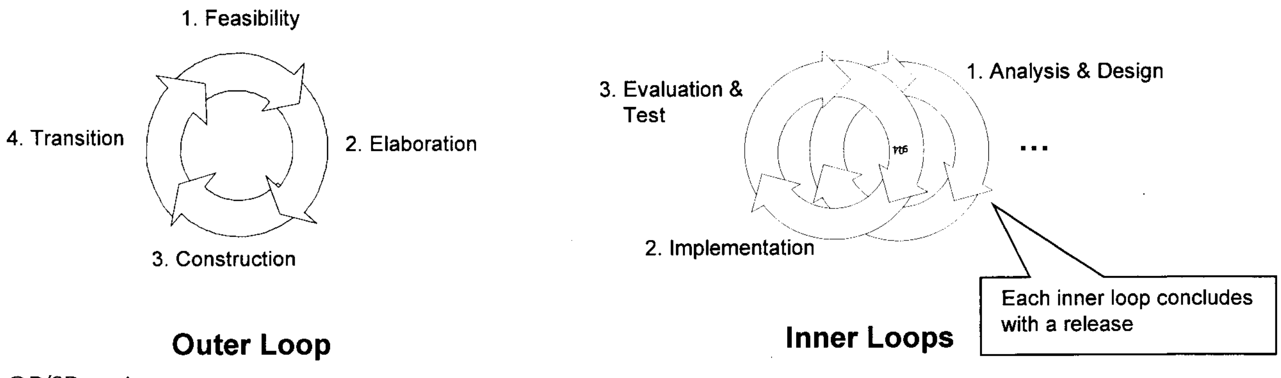

Figure 1: The outer loop progresses with iteration cycles of the inner loops

In the following discussion the outer and inner loop phases are described serially and thus have somewhat the flavor of a classic waterfall lifecycle model. The reader should not assume this to be the case. The lifecycle being espoused here is both incremental and iterative. Management and systems activities tend to be incremental while the software activities are primarily iterative. Although the management and systems activities will be incremental they will not be waterfall activities in the strict sense of this term. Few if any software projects ever truly follow a classic waterfall lifecycle process even when claiming the waterfall lifecycle as their development model. Software development is usually somewhat more chaotic with jumps back and forth between earlier and later phases in the lifecycle as the project progresses. There is often a lack of clear demarcation between phases. As personnel move from one phase to another in the lifecycle the activities associated with the first phase taper off and those associated with the next increase. In successful projects however the back and forth movement between phases will dampen out as the project moves closer and closer to a delivery. The MDS team expects to move back and forth between development phases (both outer and inner loops) as is necessary to produce the negotiated products for their customers. BFC projects at the lab are being encouraged to increase parallelism in their development efforts. This may also increase movement between development phases.

The words "MDS team, MDS system engineers, and MDS developers" are all used loosely in the following discussion of the MDS lifecycle. It is assumed that MDS system engineers are any system engineers (including software system engineers) from the mission customer, X2000 1st Delivery team and the MDS project who are working on MDS systems tasks. Similarly MDS software developers are any software engineers from the mission customer, X2000 1st delivery or MDS project who are performing MDS software development tasks. MDS team refers to any personnel directly supporting the MDS effort whether or not that person is funded by MDS, a customer project or some other aligned organization on lab.

6.3.1 Outer Loop

The outer loop consists of four phases: feasibility, elaboration, construction and transition. Outer loop activities are primarily management and systems and software systems engineering activities. The MDS project will complete one circuit of the outer loop for each formal delivery it makes to a customer. A brief discussion of the outer loop phases follows.

6.3.1.1 Feasibility

During the feasibility phase of the MDS lifecycle MDS management and system engineers negotiate with a potential customer to determine if the MDS mission data system product is appropriate for that customer mission's software needs. An initial agreement (MOU) is worked with the potential customer. If there is something technically new or challenging associated with the customer's mission the MDS team may do some high level proto-typing of the item as a proof of concept effort.

6.3.1.1.1 Feasibility Inputs

The customer project provides MDS with early versions of some or all of the artifacts listed below (initial mission requirements from the customer are required at this step in the process). MDS provides the potential customer with an overview of the MDS architectural approach and capability examples from a previously released MDS Capabilities Catalog (if one exists). MDS Capabilities Catalog will provide the customer with examples of the capabilities that the MDS software will support (obviously this step will be skipped for the first MDS development effort).

- Customer mission requirements (level 1, 2 & 3 project mission requirements)
- Customer mission plan with mission science objectives (if available)
- Customer Mission Operations Concept (if available)
- Set of customer key mission operations scenarios (if available)
- MDS Control Architecture Design Document
- MDS Capabilities Catalogs from past customers

6.3.1.1.2 Feasibility Process

MDS management, system engineers and customer(s) work to understand the customer's technical and programmatic needs. Both sides review initial customer input products. MDS management and system engineers begin a delivery plan with the customer. MDS management may produce an MOU documenting initial high level agreements between MDS and the customer. MDS development processes, infrastructure and tools are reviewed by the MDS team and adjusted, updated, or replaced as needed for the upcoming development effort. Updates and/or changes to MDS processes, infrastructure and tools are less likely to be disruptive if they are made at the beginning of a development cycle rather than in the middle of it. Making adjustments at the beginning of the development cycle does not preclude a change during other phases of the lifecycle if project management and the team determine that the change is crucial for the success of the project. As of this writing the MDS team has already begun a transit of its outer loop for its OP/SP mission customer. Since this is the very first time though the MDS lifecycle the process may not work exactly as described here. The MDS team is currently in the process of refining its processes, infrastructure and tools and this refinement is likely to continue into other phases of the lifecycle.

6.3.1.1.3 Feasibility Products

The following items will be produced in this phase:

- Documentation summarizing results of MDS review of initial customer requirements
- An initial draft of Delivery Plan for the customer that is a first cut at identifying the work activities, products to be produced and the schedule for their development, delivery and maintenance
- Optional customer MDS MOU separate from delivery plan (only if required by the customer but this should be not mandatory as all MDS customer agreements can be successfully captured in the MDS Customer Delivery Plan)
- MDS Software Management Plan (or updates to plan if plan already exists)
- MDS Information Management Plan and MDS Configuration Management Plan (or updates if plans already exist)
- MDS Verification Process Guidelines document (or updates of document already exists)
- MDS Software Quality Assurance Plan (or updates if plan already exists)
- Documentation of expected software tool and equipment usage for the current effort (or updates as appropriate to existing documentation)

6.3.1.1.4 Feasibility Verification

The following verification activities should take place (these may be informal peer reviews):

- Review of customer inputs – MDS management, system engineering, & development team leads
- Review of MDS process documents (or updates to documents as appropriate) – MDS team
- Review of existing software design and coding guidelines (or updates to documents and IOMs as appropriate) – MDS team

6.3.1.2 Elaboration

During the elaboration portion of the outer loop cycle the MDS team does requirements analysis, systems analysis, object analysis, and an initial architectural design for the evolving MDS product. The MDS team must complete enough system and software system analysis and system preliminary design during this phase to hold a Preliminary Design Review with customer(s) (note this review can be either a formal PDR and/or a series of PDR level peer reviews). The MDS Project Manager will determine both the review approach and when review(s) are to be held. The review may be held during elaboration or at its end. If MDS is negotiating delivery plans with multiple customers in parallel MDS may hold combined reviews for all of its customers.

6.3.1.2.1 Elaboration Requirements Analysis

During the requirements analysis portion of the elaboration phase of the MDS product lifecycle MDS management and system engineers continue negotiations with customers to flesh out the MDS customer delivery plan. Customer input requirements, mission scenarios and operations concepts are analyzed and transformed into MDS requirements work products.

6.3.1.2.1.1 Requirements Analysis Inputs

The customer project provides MDS with the following system engineering artifacts (or updates if previously provided). These are as follows:

- Customer mission plan including mission science objectives
- Customer mission operations concept (IOM briefly describing project mission operations assumptions and operations plan. IOM should include customer assumptions about ground system tools, resources and personnel support)
- A set of key operations scenarios for the mission (in textual format or as UML use cases)
- Customer mission requirements update if customer thinks this is needed

6.3.1.2.1.2 Requirements Analysis Process

MDS systems engineers review customer inputs as well as the current MDS project vision to develop a set of textual capability statements that functionally describe the MDS product to be developed for the customer(s). The team then uses these statements in conjunction with the other customer inputs (mission scenarios in particular) as the starting point for all of the following:

1) identifying MDS system level use cases and their associated actors;
2) identifying and characterizing external events that affect the system;
3) defining behavioral scenarios that capture dynamics of the system; and
4) identifying required constraints on the system

As part of use case analysis the MDS team identifies an initial set of goals for the system. The team examines use cases to see if they map to a activity that could be a goal. The team must ask itself what state determination (i.e. what variables) would be required to achieve closed loop control if the use case became a goal. Not all use cases will map to goals. For use cases that do map a goal however the team should identify the goal's state variables and expected method of state determination and control. This information could influence system analysis decisions with respect to hardware/software and flight/ground trades in the next step of the process.

MDS team must identify and/or update design patterns that will be applied to the evolving software system in the design phases of the lifecycle. Since it may take the team a fair amount of time and effort to determine what patterns to apply discussions to determine this should begin as early as possible. Design patterns may be captured several places. They may be captured in OOA/OOD models, in MDS design notes and in adapter guides. Similarly the team must develop or update coding standards. These should be in place prior to implementation. Since developing coding standards can involve a significant amount of work it is recommended that this process be started here or in the feasibility phase of the lifecycle so that it is complete prior to implementation.

The MDS Verification Engineer begins a validation test plan for the customer.

As the project moves into the elaboration phase of a delivery cycle the MDS team continues to refine the customer's development plan. As the team develops a better understanding of the technical effort the Project Deputy Manager, MDS system engineers and MDS team leads work schedule, budget and workforce details. MDS team leads begin planning their inner loop activities and work the coordination of those activities with each other and the rest of the MDS team. The MDS Project Deputy Manager incorporates team lead planning efforts into the delivery plan(s) for the customer(s). Any budget and schedule shortfalls that may result in capability cutbacks are negotiated with the customer(s) as part of this process.

6.3.1.2.1.3 Requirements Analysis Products

The following MDS system level items will be produced in this phase:

- Expanded Delivery Plan - MDS management, MDS team leads, MDS system engineers, customer management
- MDS Capabilities Catalog (includes textual capabilities statements captured in DOORS, and appendices containing UML use case diagrams, use case scenarios in sequence diagram format. These UML products will be developed in Rhapsody or other TBD OOA/OOD tool) – MDS systems engineers, customer system engineers and software domain leads
- Initial goal set including expected state determination and control mechanisms for goal achievement and maintenance
- IOM detailing the processes and templates that will be used for documenting messaging within the system (work products resulting from this process will replace traditional command, telemetry and fault protection/autonomy ICDs) (or updates as needed to existing IOM)
- Validation Test Plan and initial set of test procedures for validating as built system against requirements (capabilities) prior to interim capabilities releases and final delivery to a customer – MDS Verification Engineer
- MDS Design Notebook that captures software design guidelines that are at a lower level than the architectural design guidelines in the MDS Control Architecture Design Document (or updates as needed to existing documents)
- Software Coding Guidelines document (or updates as appropriate)

6.3.1.2.1.4 Requirements Analysis Verification

The following verification activities will be held:

- Peer Review of Capabilities catalog – MDS team and Customer
- Review of Validation test plan – MDS and customer system engineers

6.3.1.2.2 Elaboration Systems Analysis

During the systems analysis portion of elaboration phase of the lifecycle MDS system engineers (this includes software system engineers) work with the organization(s) that provide the hardware platform(s) upon which MDS software is to reside to specify the functionality and behavior of the overall system (both hardware and software). Wherever possible MDS systems engineers will participate in hardware selection decisions, hardware/software functionality trades and flight/ground/operations trades in order to guarantee compatibility between MDS software architecture, customer mission needs (for development, test, and flight) and underlying hardware capabilities.

6.3.1.2.2.1 Systems Analysis Inputs

Customer projects provide MDS with their system engineering artifacts (or updates if previously provided). These are as follows:

- Customer spacecraft definition (S/C functional and physical block diagrams, S/C hardware specifications such as switch lists etc., FMECAs, customer single point failure policy and exceptions, hardware fault protection, detection, and recovery approach, hardware/software and flight/ground partitioning decisions)
- Statemate models of customer spacecraft and mission activities if these have been developed by the customer as part of their system analysis and design trade studies
- Ground system definition (functional and physical block diagram of existing ground system)

6.3.1.2.2.2 Systems Analysis Process

After system level requirement analysis MDS and customer system engineers perform systems analysis tasks. Systems engineers from all the aligned teams should work together to characterize system functions, behaviors and activities. This includes identifying large-scale organization units within the system, building and analyzing complex behavioral specifications for the organizational units, partitioning system-level functionality into software and hardware and (optionally) testing the behavior of the system with executable models. The resulting system and requirements models form the system specification. At the end of this process the MDS validation team can add a set of test procedures for validating the system design to the MDS Validation Test Plan.

6.3.1.2.2.3 Systems Analysis Products

The following MDS system level items will be produced in this phase:

- Expanded Delivery Plan - MDS management, MDS team leads, MDS system engineers, customer management
- Set of textual and diagrammatic products that describe and bound what MDS is producing for the customer:
  - MDS customer spacecraft (deployment and component diagrams, textual description)
  - MDS customer ground system (deployment and component diagrams, textual description)
  - MDS customer operations concept (statecharts, activity diagrams, sequence diagrams, etc, should include some MDS mission scenarios that MDS will use to validate system)

  – MDS system engineers (includes X2000 first delivery personnel), customer system engineers and domain experts
- (Optional) Statemate models of aspects of the MDS customer spacecraft that explore some of the dynamics of the system (could consist of updated Statemate models from customer missions when/where applicable) – System engineers for customer mission, TBD
- MDS/X2000/Customer design trade documentation as needed. Should be IOMs or other written material containing write up of specific design trade studies and recommended solution (hardware software trades, etc) – MDS and customer system engineers, MDS team leads and their software implementers

6.3.1.2.2.4 Systems Analysis Verification

The following verification activities will be held:

- Peer Review of MDS customer spacecraft, ground system and associated operations concept
- PDR and/or associated peer reviews could be held at the end of system analysis or the object Analysis effort
- Review of validation test procedures – MDS and customer system engineers

6.3.1.2.3 Elaboration Object Analysis

As the system specification begins to solidify the MDS team can move to object analysis. The team uses the black box functional and behavior views of the system developed during requirements and systems analysis as the starting point for developing a white box object view of the system. The team will apply object identification strategies, including the application of any essential architectural patterns, to their system specification products to create an object model of the system.

6.3.1.2.3.1 Object Analysis Inputs

MDS is the main supplier of inputs for this analysis effort.

- MDS products from Requirements and Systems Analysis

6.3.1.2.3.2 Object Analysis Process

After a system specification exists (it may not be complete) software knowledgeable MDS systems personnel (both MDS system engineers and MDS development team leads) can begin object analysis. As Douglass points out this process is fundamentally different from the two other analysis processes preceding it and represents a non trivial transformation of the system specification (the team must migrate a black box functional view of the system to a white box object oriented one). Members of the MDS team will use knowledge gained during requirements and systems analysis to develop objects, classes, their organizational units (packages, nodes and components), and associated dynamic behavior models for the classes. Douglass notes that during object analysis the team captures only those objects and classes that are "essential to all possibly correct solutions" for the system.

MDS engineers participating in object analysis will develop object and class, and collaboration diagrams in UML using the project OOA/OOD case tool. The team will produce both static and dynamic object models of the system. The object structural model is represented with class and object diagrams, domain diagrams and component diagrams. The object behavioral model is represented with statecharts, activity diagrams, sequence diagrams, collaboration diagrams and timing diagrams. The team applies essential architectural patterns to the model during this effort.

During the entire analysis phase the MDS architect leads the rest of the team in identifying architectural and lower level design patterns that will be applied during object analysis and in the next phase of the lifecycle, object design. Architectural design patterns are documented in an MDS architecture design document. Lower level design patterns may be documented in diagrams within the case tool, in textual notes, in APIs, etc.

6.3.1.2.3.3 Object Analysis Products

The following MDS system level items will be produced in this phase:

- Baselined Delivery Plan identifying scope of MDS work effort, extent of customer participation, products to be produced, joint tools and processes to be used and the development schedule (will reference other MDS artifacts such as Capabilities Catalog, customer system specification, object model that will be put under configuration control as part of the baselined plan) - MDS management, MDS team leads, MDS system engineers, customer management
- Object Analysis Model represented in a set of diagrams from case tool (class and object diagrams, domain diagrams, component diagrams for object structural model and statecharts, activity diagrams, sequence diagrams, collaboration diagrams for object behavioral model, timing diagrams) – MDS team leads, MDS architect, MDS system engineers
- Updated MDS Control Architecture Design document – MDS architect, MDS system engineers and development team leads
- Set of IOMs documenting initial high level interface agreements between 1) flight and ground parts of MDS system within same software domain, 2) across software domains within either the flight or ground parts of the system, and 3) developers and test environment team (captures test capabilities and capability phasing required of MDS Test Environment team) – MDS development team leads (more detailed agreements for software interfaces will be captured later in process as software APIs, etc)

6.3.1.2.3.4 Object Analysis Verification

The following verification activities will be held:

- Peer review of updated MDS Control Architecture Design Document– MDS team
- Peer review of MDS Design Notebook – MDS team
- Peer review of IOM interface agreements – MDS development teams involved, customer engineers with related domain responsibilities
- Peer Review of initial object model of system
- Review of MDS customer delivery plan where MDS and customer hold commitment review and sign off of the MDS customer delivery plan – MDS and customer management

6.3.1.2.4 Elaboration Architectural Design

During this part of the development effort the MDS team applies architectural patterns to their object model. This starts out as a system level effort. MDS system engineers and development team leads apply top level architectural design patterns to the object model that are not software specific. However many of the design patterns to be applied to the model will be software specific and must be applied by software developers who understand the trade space. Once the top level patterns are applied and the major software groupings identified, they will be parceled out to the development teams for further refinement within inner loop cycles. It is expected that the software groupings to emerge will more-or-less follow the MDS team organization that is currently in place. It is possible that some team re-alignments will occur as a result of this initial system level application of design patterns.

The chief inputs for the architectural design process are the outputs from the object analysis process:

- Object Analysis Models (class and object diagrams, domain diagrams, component diagrams for object structural model and statecharts, activity diagrams, sequence diagrams, collaboration diagrams for object behavioral model, timing diagrams)
- MDS Control Architecture Design document
- Updated MDS Design Notebook

6.3.1.2.4.1 Architectural Design Process

Once the MDS team has identified the logical model of a system they will begin to apply design decisions to that model to optimize certain features within it. This process will begin the transformation of the logical system model into an architectural design model. The team will apply MDS architectural patterns from MDS Control Architecture Design Document to the logical model. The team will produce class and object diagrams, component diagrams, statecharts, activity diagrams, sequence diagrams, collaboration diagrams and timing diagrams. If the design model modifies the team's understanding of the underlying system then the team will update the analysis model to reflect their new understanding.

As the team evolves the system architectural design the team will produce textual descriptions of that design. These will be incorporated into the latest version of the MDS Control Architecture Design document. For instance, after the application of control architectural design patterns to their specific domain area within the MDS system an MDS development team would document the resulting design for their domain. This description would identify domain states and the state variables associated with them, discuss how state determination and state control occur within the domain and identify the models to be used (algorithms or whatever is appropriate for the domain at hand) for the achievement of domain state determination and control.

6.3.1.2.4.2 Architectural Design Outputs

The outputs of this phase are primarily modified object analysis models that have become the initial object design models for the team.

- Object Analysis Models (class and object diagrams, domain diagrams, component diagrams for object structural model and statecharts, activity diagrams, sequence diagrams, collaboration diagrams for object behavioral model, timing diagrams)
- Updated design descriptions in MDS Control Architecture Design document

6.3.1.2.4.3 Architectural Design Verification

6.3.1.3 Construction

During the construction portion of the project life cycle a majority of the project activities take place within the inner loop cycles. Outer loop activities during this phase in the lifecycle consist largely of management and system engineering monitoring, integration, and test activities. One very important management activity during this part of the lifecycle is the management and resolution of problem reports and requests for changes to the software. Details of the problem reporting and change control processes are described in the MDS Software Quality Assurance Plan and/or the MDS Configuration Management Plan.

6.3.1.3.1 Construction Inputs

These are primarily the systems engineering products from the previous phase of the outer loop as well emerging software design products from inner loop activities (see inner loop descriptions).

6.3.1.3.2 Construction Process

MDS management monitors the progress of development teams as they transit their inner loop development cycles. Management reports to software development status to JPL management and customers, and management works contingency plans with the development teams and customers when there are capability or schedule short falls. MDS system engineers provide coordination and monitoring support to the development teams. The MDS Verification Engineer (with participation from the entire MDS team) evaluates emerging software products for their compliance with delivery requirements and operations scenarios. The MDS Quality Assurance Engineer monitors the software development effort for compliance with project quality assurance objectives. MDS system engineers coordinate the implementation and test of cross cutting capabilities being developed by the software development teams.

6.3.1.3.3 Construction Products

Systems products for this phase of the lifecycle are primarily test and delivery products. When an MDS system is ready for an interim release (capabilities releases) or final delivery to a customer the MDS project will hold an interim release or a delivery review (RDR).

The following system level products will be produced:

- Release Description Document (interim for informal releases and final for a customer delivery)
- Command/Telemetry/Fault Protection/Autonomy Design Documents
- MDS User's Guide/Operators Manual for the release or delivery
- Validation test procedures and test results for the release or final delivery
- Electronic media containing the complete development environment
- Deltas to the customer Delivery Plan in response to changes of scope resulting from negotiated change requests, problem report dispositions

6.3.1.3.4 Construction Verification

The following verification activities will be held:

- Release or Delivery SRCR
- Peer review of Validation test procedures
- Independent customer acceptance tests on delivered product

6.3.1.4 Transition

This is the period subsequent to an MDS final delivery to a customer. The MDS team supports customer adaptation of the MDS product and some level of bug fixes.

6.3.1.4.1 Transition Inputs

- Delivered MDS product

6.3.1.4.2 Transition Process

Once an MDS delivery is made MDS will provide customer training, product installation and product maintenance support (bug fixes and adaptation help) to the receiving customer as negotiated in the delivery plan. MDS will continue support to the customer for the length of time and to level originally established in the customer delivery plan. The customer will use the MDS problem reporting system to report problems with the MDS product to MDS after delivery. All post delivery fixes and upgrades will be negotiated between MDS and the customer according to the maintenance agreements specified in the MDS customer delivery plan.

6.3.1.4.3 Transition Products

- IOM documenting delta delivery agreement with customer if fix to delivered software is to be made
- Delta Release Description Document if a post delivery upgrade/fix is made to the delivered product
- Problem failure reports (PFRs or ARs if TMOD system is used) and Software Change Requests (SCRs or CRs)
- Electronic media containing partial or complete development environment(depends on fix)

6.3.2 Inner Loop

An inner loop consists of the following processes: analysis and design, implementation, evaluation and test. An MDS development team will complete one circuit of the inner loop for each set of capabilities the team implements as a group. Inner loop developments will be coordinated across development teams when there is functional coupling between teams.

6.3.2.1 Analysis and Design

Object analysis started as a system level activity during the elaboration phase of the lifecycle (see outer loop discussion above). The implementation teams will expand and refine the initial analysis and architectural design models as they initiate their own architectural, mechanistic and detailed design activities.

6.3.2.1.1 Analysis and Design Inputs

- Object Analysis (logical) Model
- Architectural Design Model
- Design optimization guidelines, and design patterns appropriate to the architectural, mechanistic and detailed design efforts of the domain specific development teams

6.3.2.1.2 Analysis and Design Process

Once MDS engineers have done the initial object analysis and architectural design models for a delivery and development team leads have scoped and prioritized their team's efforts with MDS management the development teams can get busy. Each team will traverse one or more cycles of their inner loop to complete their portion of the evolving MDS product. A development team may revisit and refine analysis and architectural design products produced in previous outer loop elaboration phases by system engineers. They apply lower level design patterns from MDS design notes to their piece of the architectural model. This will include applying task scheduling policies, memory management policies, error handling policies, inter-processor communication protocol decisions, and various other design decisions to the evolving design model. The team will produce class and object diagrams, component diagrams, statecharts, activity diagrams, sequence diagrams, collaboration diagrams and timing diagrams. If the design model modifies the team's understanding of the underlying system then the team will update the analysis model to reflect their new understanding.

In addition to architectural design the development team does mechanistic and detailed design. During mechanistic design the development team refines object collaborations. The team may add new objects and classes to serve as "glue" for existing object collaborations. During detailed design the team expands the internal structure and refines the behavior of individual classes within the model.

6.3.2.1.3 Analysis and Design Products

- Updated capabilities catalog including updated/expanded use cases and scenarios – System Engineers, Team leads, Developer
- Updated Analysis model (if needed)Design Model updates (if needed) – System Engineers, Development Engineers
- Mechanistic Design Model (class and object diagrams, component diagrams, sequence diagrams, collaboration diagrams, timing diagrams) - Developer
- Detailed Design Model (object model, statecharts, activity diagram, pseudocode) - Developer
- Unit test plans – Development Team leads, Developers
- Updated(fleshed out) ICDs and/or software APIs – Developer, Development Team lead, System Engineers
- Preliminary Design Description Documents (for commands, telemetry, fault protection/autonomy interfaces – if they can be created at this point in the development process) – Developer, System Engineer

6.3.2.1.4 Analysis and Design Verification

The following internal MDS reviews will be held during this phase:

- Developer Design Reviews (peer review of software design including all products listed above produced by the developer for this phase)

6.3.2.2 Implementation

6.3.2.2.1 Implementation Inputs

Inputs are the products from the detailed design effort in the previous phase.

6.3.2.2.2 Implementation Process

During this phase the development team implements its proposed design. (Code generation mechanism is currently TBD (either manual or automatic)). The resultant code is unit tested and then integrated with existing code that makes up the rest of the emerging MDS product. Depending on what the code does and where it fits in the merging MDS product there may be multiple levels of integration required before it is completely merged with the rest of the MDS product (an MDS goal is to keep this process from becoming onerous). It is the job of the MDS integration engineer to lead and to coordinate integration activities across multiple development teams, across multiple development environments, across flight, ground, and test interfaces and across the MDS TMOD interface.

6.3.2.2.3 Implementation Products

The following products will be produced during this phase:

- Source code
- Compiled object code
- Unit test procedures and results
- Linked (integrated code)
- Tested, integrated code (integration may be layered)
- Updated ICDs and software APIs (if necessary)
- Updated/expanded Design Description Documents (for commands, telemetry, fault protection/autonomy interfaces - should be auto generated from comments in the code)

6.3.2.2.4 Implementation Verification

- Source code review
- Integration test procedures and summary of results for end-to-end system integration

6.3.2.3 Evaluation and Test

6.3.2.3.1 Evaluation and Test Inputs

An integrated MDS product is the input for acceptance testing.

6.3.2.3.2 Evaluation and Test Process

At set points in the development process when certain capability thresholds are reached the integrated MDS product is turned over to verification team for regression and for acceptance testing. Once this team completes its validation of the product it is released informally (interim release) or it is delivered to the customer (final delivery). See the process description under the construction phase of the outer loop above for more details. MDS expects customers to participate in acceptance testing of interim products. Early test involvement with the evolving product will help customers assess the applicability of the product to their own project needs before it is complete. This will allow customers to file problem reports and change requests in time to influence the contents and capabilities of the final delivery product (details to be worked – there will be checks and balances to this process).

6.3.2.3.3 Evaluation and Test Products

See the product section under the construction phase section of the outer loop.

6.3.2.3.4 Evaluation and Test Verification

See the verification section under the construction phase section of the outer loop.

7 CUSTOMER BUILD PLANNING

MDS will develop a delivery plan with each customer. See description below.

7.1 Incremental Build Identification

Details of build identification are described in the MDS Configuration Management Plan.

7.2 Functionality Inclusion Order Rationale

The MDS development approach is incremental but it also supports continuous integration. In order to meet both objectives the MDS team will implement a complete but thin version of the end-to-end mission data system as early as possible in the current software development lifecycle. This will establish and exercise end-to-end interfaces in the MDS system. Subsequent software development efforts will then focus on flushing out functionality within software components attached to the interfaces. Highest priority software capabilities will be implemented in earlier incremental builds of a delivery cycle whereas lower priority items will be implemented later. MDS will make software implementation priority decisions according to several (possibly competing) guidelines. Customer delivery schedules will drive priority. Since X2000 is the first MDS customer and we need to validate hardware before loading application software on it X2000 software test requirements will be given higher implementation priority than OP/SP customer mission requirements. New and/or difficult functions will be implemented early. The concept of Goal Achieving Modules (GAMs) is central to the MDS architecture. This concept is new to flight software development, and may be difficult to implement successfully. MDS should proto-type this concept as early as possible in the software as part of a risk mitigation and proof-of-concept effort. The need to coordinate cross-cutting MDS products with the rest of TMOD may also drive implementation priorities. MDS customer development plans will include schedules that detail capability inclusion order and coordination across MDS development teams.

MDS will not implement all new and/or possibly difficult to implement functions at once. New and/or hard to implement capabilities may be phased so that possible problem areas are tackled by the MDS development teams serially.

7.3 Build Delivery Schedule and Functionality Coordination

Build delivery schedules are part of the customer's delivery plan. MDS system engineers coordinate the inclusion of capabilities that cross cut multiple MDS development teams and/or external organizations such as TMOD.

8 SOFT WARE DOCUMENTATION

The MDS Project Library has a section for MCDL documents. The term "document" is used loosely. Work products such as electronically maintained design files generated from the MDS case tool Rhapsody are considered "documents" and will be part of MDS MCDL.

8.1 General

Work product templates, naming conventions, and version control are discussed in the configuration management section of the MDS Information Management Plan and/or in the MDS Configuration Management Plan. See the MDS electronic library for details. All MDS project documents are in the MDS Project Library. The library is accessible through the MDS Project Home Page.

8.2 Development Plans

8.2.1 Software Management Plan

This document is the MDS software management plan. It follows JPL software management plan guidelines including compliance to ISO9000-3 standards applicable to software development. See the ISO compliance matrix in appendix C.

8.2.2 Delivery Plan

For each formal delivery to a customer MDS will develop a delivery plan. This document will serve as a contract between MDS and the customer. The delivery plan will define deliverables and the schedule for their completion (customer to MDS and MDS to customer). The plan will prioritize the development of software capabilities according to their criticality to the delivery and to mitigate MDS and/or customer technical and programmatic risks. The plan will identify the processes and personnel to be jointly shared by MDS and customer mission in order to achieve plan objectives. The plan will identify MDS training and licensing responsibilities. The plan may also identify pieces of TMOD external to MDS that will be integrated by MDS as part of the end to end mission data system being developed by MDS for the customer. The plan will identify agreed upon fall back measures to be applied by MDS in the event that promised capabilities cannot be developed as originally envisioned or that schedules will not be met. The plan will call out primary metrics to be used by MDS to report development status to the customer on a regular basis. Finally the plan will document any agreements between MDS and the customer with respect to the MDS development processes and products that are a departure from the MDS software processes and products described within this and other MDS process documents. Negotiation of non-standard processes and products is discouraged but may be necessary in certain instances. A customer delivery plan should be in draft form no later than the MDS PDR for the customer and finalized prior to MDS the CDR for the customer.

Since MDS software is developed incrementally MDS will produce updates to a customer delivery plan at the start of each new build cycle (6 month intervals). The update will include requirements cleanup/expansions for the capabilities to be implemented in the build. The customer project will participate in the update by supplying updates to their MDS requirements inputs. The customer updates should be primarily to address the capabilities being developed in the next build cycle. Additionally MDS and the customer will discuss problems and/or shortfalls of the previous build(s) and agree upon what fixes or modifications should be included in the new build to correct the problems/shortfalls of the previous build.

8.2.3 Phase Plans

For each delivery cycle there will be multiple inner loop cycles (inner loop iterations) across the MDS development teams. In general, within a delivery cycle the phase cycles amongst the development teams will run in parallel but may not always be synchronous. Within a delivery cycle a single development team will experience multiple serial phase cycles (inner loop iterations). At the start of an inner cycle the development team(s) will develop a detailed phase plan for that iteration (a higher level version of this plan will have already been included in the delivery plan or its update). The phase plan expands the initial software development schedules and products specified in the delivery plan(s) to a level detail sufficient for the development team to plan its work. Where phase cycles of two or more development teams need to be synchronous (they are developing coupled capabilities) this will be addressed in the phase plans of both teams. Phase plans delineate the elaboration of any interface agreements between the team and groups external to it (other MDS development teams, MDS test environment team, external portions of TMOD, customer(s), etc.) that must be concluded as part of the loop cycle. Phase plans identify all test capabilities that will be needed by development team for the loop cycle. Additionally the plan should reference an MOU developed between the team lead and the MDS Test Environment lead (if needed). This MOU should describe in detail exactly what test software capability the MDS test environment team will produce to aid the development team's testing and when the test soffware will be available for use.

8.3 System Analysis Documents

8.3.1 Requirements Documents

MDS will capture top level architectural requirements and design descriptions in the MDS Control Architecture Design Document.

MDS will capture functional requirements as textual capabilities statements in a Capabilities Catalog, as textual statements in DOORs requirement database tool and as use cases and sequence diagrams in an OOA/OOD case tool. Note that capabilities will have the flavor of traditional mission level 2, and level 3 functional requirements. A baselined version of MDS capabilities associated with a particular customer's delivery plan will be available electronically in the MDS library. The MDS team will translate and expand textual capabilities statements into UML use cases, scenarios and activities diagrams in their OOA/OOD case tool. Eventually MDS requirements will be captured primarily as UML use case and scenarios. These will be exported into a DOORS database so that they can be linked to customer textual requirements in DOORS. A baselined set of textual capabilities and associated UML files for a specific customer delivery will be available on line at the MDS web site.

8.3.2 Reference Spacecraft Definition

This is a collection of artifacts (textual and diagrammatic work products from systems engineering) that collectively constitute the definition of an MDS reference spacecraft. MDS will produce a reference mission system as an MDS implementation example for a mission customer. The MDS mission will look something like the customer mission but it will be not be exactly like the customer mission. These artifacts will be included in an appendix of the customer's delivery plan

8.3.3 MDS Operations Concept IOM

This is an IOM that MDS produces documenting the operations capabilities of their product for a specific customer. It will be included in the customer delivery plan.

8.4 Design Documentation

8.4.1 Architectural Design

MDS architectural requirements in conjunction with MDS capabilities will give rise to MDS architectural design specifications. Textual descriptions of these specifications will be captured in the MDS Control Architecture Design document.

8.4.2 Detailed Design

Detailed design will be done by the MDS software developers at the start of an inner loop iteration. Detailed design information will be captured as brief textual descriptions, interface specifications (APIs) and in UML diagrams in the MDS OOA/OOD case tool.

8.4.3 Interface Control IOMs/Documents/APIs

The MDS team infrastructure has been constructed to minimize the number of interface agreements required between development teams. The form of inner (intra) team agreements for software interfaces that are completely located within one MDS technical domain, implemented by a single development team, have no external interfaces and need not be adapted by other developers are up to the discretion of the team. Software interfaces that cross development teams or have flight ground interfaces, or will be adapted customer uses must be documented appropriately (API or ICD). The draft API or ICD should be presented at the appropriate development team design reviews and must be finalized before the corresponding software capability is integrated into the end to end MDS system.

8.4.4 Command/Telemetry/Fault Protection/Autonomy Design documentation

MDS will generate state, goal, and telemetry dictionaries. An MDS goal is to develop a process by which these documents can be auto generated. These documents will provide multiple views of their associated data including a ground user view and a flight software view. MDS will document fault protection design decisions (may be part of state database information). Additionally operational constraints (flight rules) will be included in these documents where appropriate. Details are TBD....

8.4.5 Data Dictionary

MDS will generate data dictionaries to describe the structure and contents of MDS ground and flight based database(s). Details are TBD....

8.5 Testing Documentation

8.5.1 Development team Unit and Integration Test Plan and Procedures

MDS developers will develop unit and development team internal integration plans and procedures. These will follow guidelines set out in the MDS Verification Guidelines document.

8.5.2 Integration Team Test Plan and Procedures

The MDS integration engineer and team will develop integration test plans and procedures. These will follow guidelines set out in the MDS Verification Guidelines document.

8.5.3 Verification Team Validation Test Plan and Procedures

The MDS Verification engineer and team will develop validation test plans and procedures for acceptance testing of the MDS product prior to delivery to a customer and/or to X2000/MDS 1st Delivery Project. The extent of this effort (number and type of customer mission scenarios to be used for acceptance testing by MDS personnel) will have been previously characterized in the MDS customer delivery plan.

8.5.4 Delivery Documentation

An informal Release Description Document (RDD) will be produced by an MDS development team at the conclusion of software iteration cycle (i.e. at the conclusion of an inner loop cycle). A formal Release Description Document will be produced by the MDS Verification Engineer with support from MDS system engineers, MDS test environment engineer and MDS implementation leads for each MDS formal delivery to a customer and/or X2000/MDs 1st Delivery Project. Release Description Documents (RDDs) will be prepared in accordance with JPL D-4007. A release description document will include an inventory of all items delivered, detailed build instructions, description of any new or changed capabilities, summary of problem reports addressed, identification of the program set being released and the applicable environment. This document is presented to the customer at the SRCR for an MDS delivery to a customer. When X2000/MDS 1st Delivery Project delivers MDS software to a mission customer it is assumed that the same process will be followed with updates being provided to the original SRCR documentation by the X2000/MDS 1st Delivery organization personnel responsible for validation of MDS software on X2000 hardware.

8.6 Operations

MDS User's Guide/Operations manuals will be produced for each MDS formal delivery. Details are TBD.

8.7 Configuration Management

MDS configuration management policies and procedures are being captured in two documents. General information management is described in the MDS Information Management Plan. Software based configuration management is described in the MDS Configuration Management Plan.

8.8 Quality Assurance

Quality assurance engineers from the JPL Quality Assurance Organization participate in MDS process definition and help to define MDS quality procedures and practices. MDS project quality assurance requirements and guidelines are documented in the MDS Software Quality Assurance Plan. MDS quality assurance engineers will monitor MDS personnel, processes and products for compliance.

9 SOFTWARE REVIEWS

Reviews are part of MDS risk mitigation and quality assurance planning. MDS will hold a number of reviews throughout the project lifecycle. There will be both formal and peer reviews at various levels within the MDS project. See the MDS PIP for details on project level management reviews. Technical reviews (formal and peer) have been identified in sections describing MDS lifecycle details in this document.

The MDS approach will be to hold peer reviews rather than formal reviews although MDS will hold a number of project wide formal reviews as dictated by the MDS PIP and JPL policies (See MDS PIP for details). MDS will hold peer reviews in conjunction with formal reviews and at specific software lifecycle milestones. Peer reviews are intended to be "work level" reviews and are primarily to help MDS management and technical leads assess progress, identify technical and management issues and establish action plans for their resolution. Peer review results will be presented at formal reviews (primarily MMRs and Quarterly Reviews). Peer Reviews will include at least the following:

- Project Process and Tool Reviews
- Preliminary Design Review for MDS systems engineering and each MDS domain area
- Detailed Software Design Review for each MDS domain area
- Code Review
- Integration test plans and procedures
- Validation test plans and procedures

Review details are covered in the software lifecycle section of this document and in the MDS Quality Assurance Plan (review criteria, etc.).

10 DEVELOPMENT TESTING

10.1 Testing Approach

Good verification processes are described in the MDS Verification Process Guidelines document. This document describes a number of processes that MDS developers and testers can follow to do test planning, test procedure design, procedure generation, and test execution. MDS teams will use this document as a guideline. In general MDS development and test teams pick and implement a subset of the procedures described in the MDS Verification Process Guidelines document that meet their specific test needs while conforming to their resource constraints. MDS teams will document their tailoring in their test plans.

Development test planning begins early in a major delivery cycle (elaboration phase of the outer loop) but most testing details are worked as part of inner loop iterations. Development leads and the MDS integration team are responsible for developing test plans and procedures for software verification activities during inner loop cycles. Test plans and procedures are presented and reviewed at detailed software design reviews. Developers do both unit testing and integration testing of their software prior to its merger with the rest of the MDS system. The MDS integration lead coördinates integration testing of merged MDS products at the end of each inner loop cycle, and at the end of synchronized inner loop cycles prior to release of the end-to-end MDS system to the MDS Verification Engineer for acceptance (validation) testing.

10.2 Testing Coordination with customers

MDS acceptance testing will be a coordinated effort. MDS expects customers to participate in this test activity either directly as members of the MDS verification team or indirectly by exercising interim MDS releases on the customer testbed or in a software workstation environment. Additionally MDS personnel and customer personnel will participate in X2000/MDS 1st Delivery Project testing of MDS software on X2000 avionics hardware.

11 ACCEPTANCE TESTING

MDS will perform acceptance testing on MDS products prior to delivering them to either X2000/MDS 1st Delivery Project or a mission customer. The MDS Verification Engineer performs validation testing including regression testing on both interim and final MDS software products once they have completed integration testing. MDS expects customers to actively participate in validation testing. Customer assessment of and feedback on the evolving MDS product in early iteration cycles is essential to the production of an MDS product that meets customer expectations. Early assessment will allow customers to negotiate fixes to and fine tune features of the evolving product.

12 QUALITY PLANNING

12.1 Recurring problem identification and correction

Details of the MDS approach for the identification and correction of recurring problems are covered in the MDS Software Quality Assurance Plan. See sections on risk management and metrics.

12.2 MDS Quality Assurance

MDS quality assurance personnel responsibilities, procedures and practices are described in detail in the MDS Software Quality Assurance Plan.

12.3 MDS Quality Record Management

The MDS project will produce at least the following quality records:

- Approved changes in requirements
- Review or verification results, including a summary of requests for action (RFAs) and the responses thereto
- Anomaly reports (problem/failure reports)
- Checks of test tools, to evaluate whether the tools are capable of verifying the acceptability of the software product under development
- Test results, with clear indications whether the product has passed or failed
- Change requests generated during development and after delivery (during maintenance)
- Presentation viewgraphs for all reviews

Details of how MDS handles its quality records, including retention times are covered in the MDS Information Management Plan.

13 CONFIGURATION MANGEMENT

MDS configuration policies and practices are described in the MDS Information Management Plan and the MDS Configuration Management Plan.

14 DEVELOPMENT STANDARDS

Standards and procedures pertaining to software configuration management activities are documented in the MDS Configuration Management Plan. Standards and procedures relating to information management and in particular documentation and control record management are covered in the MDS Information Management Plan.

Software design standards are covered in the MDS Control Architecture Design Document, in various informal notes stored in the project's electronic library and in software APIs.

Coding standards are documented in the MDS Coding Standards document.

15 SOFTWARE METRICS

15.1 General

The MDS development process and the products developed within it will be measured to accomplish the following primary goals:

- Identify and control risks to successfully completing and operating the products developed by MDS
- Improve the efficiency of the development process by identifying redundant, missing, or incorrect activities and changing the process definition to correct these problems

Detailed goals, descriptions of the measurements that will be taken, analysis of these measurements, and application of the results to the controlling MDS products and the development process itself are described in the MDS Software Quality Assurance Plan. As a matter of policy, measurement should be minimally intrusive to the development activity, consistent with achieving the goals defined in the MDS Software Quality Assurance Plan. Also as a matter of policy, any proposed measurement activity requiring the expenditure of MDS resources must be associated with a specific goal or set of goals, and those goals must be shown to be directly related to satisfying the primary goals defined above.

16 SOFTWARE TOOLS, METHODS, AND ENVIRONMENTS

16.1 Overview

MDS software will be developed in several environments. Tools will vary with the environment being used. MDS analysis, design and software implementation tools are described in on line documentation at the MDS web site or in the MDS electronic library. Project constraints on the use of these tools are also described in on line documentation at the MDS web site or in the MDS electronic library. Many of the tools in use by the MDS team have a help capability built into them.

The MDS Configuration Management Plan and/or the MDS Information Management Plan discuss configuration management procedures for the various MDS tools.

16.2 Software analysis and design

MDS software analysis and design tools will be hosted on FST servers. The MDS team will run these tools from their local computer whether a PC, MAC or UNIX machine.

16.3 Flight and Ground Software

MDS is a unified flight ground mission software system. Source code developed for number of the MDS domains will be instantiated in both the flight and ground systems of a mission customer. MDS execution and planning, MDS data management, and MDS data transport are three MDS software domains where significant pieces of source code will be instantiated in both the flight and ground software systems of a mission. MDS will not completely replace legacy JPL ground system software elements in the near future. The MDS team will integrate MDS developed ground software with legacy ground system software.

16.4 Test Environment

Initial MDS software development will be on PC, MAC and UNIX machines. Unit testing and initial integration testing will occur within this local development environment and/or on a UNIX workstation in which all interfacing hardware is simulated. Integration and test of the end-to-end MDS system for the first MDS mission customer will be on the PT-MDS, on the X2000/MDS 1st Delivery Project PT-FDP, and on the EM-FDP. PT-MDS will contain commercial hardware that functionally resembles the X2000 flight hardware to be delivered to MDS customer missions. This will enable MDS developers to integrate and test their software with some real spacecraft hardware. However many parts of the MDS reference spacecraft will continue to be simulated. The PT-FDP and EM-PDP testbeds will have more hardware and/or hardware that is more flight like (engineering instead of proto-type hardware models). MDS software for the first MDS mission customer will be exercised on these other testbeds as they become available. Depending on the delivery plan negotiated with a customer MDS and the customer may do some shared testing of MDS on a customer testbed with more customer specific hardware in place such as attitude control sensors. The goal of this testing would be to exercise MDS software with hardware would otherwise only be simulated in the MDS testbed and/or X2000/MDS 1st Delivery Project testbeds.

17 PROCUREMENTS

MDS procurement policies and procedures are covered in the MDS Project Implementation Plan.

18 TRAINING

MDS will provide some training to customer teams in MDS software development processes and tools prior to delivery of MDS software to the customer. Training details and responsibilities will be delineated in the MDS customer delivery plan. It is vital that MDS customer development teams be well trained in MDS methodologies and processes prior to beginning their own development effort.

19 SOFTWARE DELIVERY & INSTALLATION

MDS delivery process details are described in the software lifecycle section above. Specific and or special installation agreements will be negotiated on a case by case basis with a customer and documented in the MDS customer delivery plan.

20 SOFTWARE MAINTENANCE

MDS will provide maintenance support to its customers after delivery. The level and duration of the maintenance that MDS supplies to a customer will be documented in the MDS customer delivery plan.

21 PLAN UPDATES

This development plan and other MDS software process documents are living documents. As the project progresses successful management of the task will require adjustments to this plan and the other project process documents. When adjustments occur the custodians of the impacted project process documents will update their document to reflect the new or modified processes. Custodians will make the updated documents available to team members and project customers in a timely fashion through the MDS Project Library.

APPENDIX A: GLOSSARY

Abstraction

Aggregation

API – Application Program Interface

BFC – Better, Faster, Cheaper

CCB - Configuration Control Board

CM – Configuration Management

CI – Configuration Item

Component -

Common Examples – Implemented non-trivial solution to a problem or area likely to be common across all MDS customers

Core Software – That part of the MDS system that a customer should not have to modify (other than perhaps tweaking parameters for some specific hardware differences such as size of memory, number of processors, etc.)

COTS – Commercial Off-the-Shelf Software

DNP – Develop New Products

DNS – Domain Name Server

EM – Engineering Module

Encapsulation

End-to-End Services -

EO – Europa Orbiter

FMEA / FMECA – Failure Modes and Effects Analysis or Failure Modes and Effects Criticality Analysis

Frameworks -

FST – Flight Software Test Bed

GAM – Goal Achieving Module

GSE – Ground Support Equipment

GUI -

I&T – Integration and Test

ICD – Interface Control Document

ISO -

MOU – Memorandum of Understanding

Object

OOA – Object Oriented Analysis

OOD – Object Oriented Design

OP/SP – Outer Planets / Solar Probe

Packages

PDMS - Project Data Management System ?

PFR – JPL institutional Problem Failure Report

PIM – Project Information Management (Plan)

PIP – Project Implementation Plan

RDD – Release Description Document

RDR – Release Delivery Review ?

RFA – Request for Action

Rhapsody – Case tool for doing OOA and OOD

Reference Examples – Implemented non-trivial solution to a problem or area likely to be unique for each customer. Provides a reference for customer implementation.

Scenario

Sequence

SESPD -

SCR – Software Change Request

SRCR – Software

SMP – Software Management Plan

SPA – Software Product Assurance

System – Integrated and complete product that is usable by missions. May be legacy or new development

TMOD -

UML – Unified Modeling Language

Use Case

WMA – Work Management Agreement

APPENDIX B: PLAN COMPLIANCE WITH JPL D-15378 AND SOFTWARE RELEVANT ISO REQUIREMENTS

	ID-15378 Requirement with ISO cross reference	MDS Compliance
1.	***A software development project/task shall be organized according to a life-cycle model that is described in the development plan in terms of:*** *[5.1] [ISO 4.4.1 guidance]*	MDS SMP documents MDS software lifecycle approach
2.	***o Phases, along with milestones and activities to be performed during each phase;*** *[5.1a] [ISO 4.4.2 guidance]*	MDS SMP describes software lifecycle phases
3.	***o Phase outputs, including any documentation;*** *[5.1b] [ISO 4.4.2 guidance]*	MDS SMP describes phase outputs
4.	***o Verification activities (e.g., reviews, demonstrations, tests) by phase.*** *[5.1c] [ISO 4.4.2 guidance]*	MDS SMP describes phase verification activities
5.	***The organization that has overall responsibility for a software development effort shall establish and maintain documented procedures for a commitment review of the development plan or proposal to ensure:*** *[5.2.1] [ISO 4.3.1]*	MDS Process Lead is holding peer reviews of MDS processes with MDS team and customers. MDS processes are documented in MDS process documents.
6.	***o Scope of work for the current delivery is adequately defined and documented.*** *[5.2.1a] [ISO 4.3.2a]*	MDS SMP calls for development of MDS customer delivery plan that details work that MDS does and products that MDS produces for a customer
7.	***o Differences between the scope of work defined in the development plan, and that requested by the customer, are resolved.*** *[5.2.1b] [ISO 4.3.2b]*	MDS Capability Catalog identifies what customer requirements MDS is committing to. Capability catalog is included as part of MDS customer delivery plan
8.	***o Responsibilities of the customer are identified.*** *[5.2.1c] [ISO 4.3.2 guidance]*	MDS SMP describes some customer responsibilities. Others will be captured in the MDS customer delivery plan
9.	***o Mutually acceptable means have been defined for dealing with changes in requirements during development, as well as correction of post-delivery defects.*** *[5.2.1d] [ISO 4.3.2 guidance, 4.3.3]*	MDS will use methods documented in MDS Information Management Plan. MDS customer agreed to processes also referenced in MDS customer delivery plan
10.	***o The resources and schedule described in the development plan are adequate to accomplish the contractual deliverables.*** *[5.2.1e] [ISO 4.3.2c]*	MDS customer delivery plan tailors development effort to meet MDS workforce and schedule constraints
11.	***A record of commitment review findings shall be maintained as part of the project/task quality record.*** *[5.2.2] [ISO 4.3.4]*	Signed MDS customer delivery plan will be maintained as MDS quality record
12.	***The developer shall have a written set of software requirements that are sufficient to satisfy customer and/or user needs.*** *[5.3.1] [ISO 4.4.4 and guidance]*	See MDS SMP. MDS develops MDS capability catalog to capture customer requirements that MDS commit to meeting. Catalog is part of MDS customer delivery plan. Catalog includes textual statements and UML use case diagrams, scenarios
13.	***Interfaces between the software product and***	See MDS SMP. MDS systems engineering products

	external software or hardware items shall be specified, either directly or by reference. *[5.3.2] [ISO 4.4.4 guidance]*	include hardware software ICDs for hardware platforms upon which MDS software executes. MDS will also identify and resolve interface between MDS software and rest of TMOD software
14.	***Software requirements, whether provided by the customer or formulated by the developer, shall be reviewed to ensure that:*** *[5.3.3]*	See MDS SMP. MDS reviews customer input requirements, customer reviews, accepts MDS capability catalog as part of sign off of MDS customer delivery plan
15.	***o the product is adequately defined,*** *[5.3.3a] [ISO 4.4.4]*	See MDS SMP. Part of process of MDS and customer sign off of MDS customer delivery plan
16.	***o ambiguities and conflicting requirements have been resolved, and*** *[5.3.3b] [ISO 4.4.4]*	See MDS SMP. Part of process of MDS and customer sign off of MDS customer delivery plan
17.	***o the requirements are stated so as to allow validation during product acceptance.*** *[5.3.3c] [ISO 4.4.4 guidance, ISO 4.4.8]*	TBD. MDS Verification Engineer reviews capability catalog and develops Validation Test Plan for acceptance testing of MDS product for customer
18	***The software requirements specification shall be subject to change control procedures, once it is baselined (e.g., completion of document review, customer approval obtained).*** *[5.3.4] [ISO 4.3.3, 4.4.4 guidance]*	Agreed to requirements are baselined after MDS customer delivery plan sign off
19.	***Approved changes in requirements shall be maintained as part of the project/task quality record.*** *[5.3.5]*	Approved software change requests maintained, revised MDS customer delivery plan maintained as quality records
20.	***A development plan shall address the following:*** *[5.4.1] [ISO 4.4.2]*	MDS customer delivery plan addresses these details.
21.	***o Overall definition of the product, as in user needs addressed, deliverables, and critical functionality.*** *[5.4.1a] [ISO 4.4.2 guidance]*	MDS customer delivery plan addresses these details.
22.	***o Scope of development work to be performed, including management and supporting activities.*** *[5.4.1b] [ISO 4.4.2 guidance]*	Project WMAs, schedules for MDS customer delivery plan, description in the delivery plan.
23.	***o Project life cycle, including:*** *[same requirement as 5.1] [ISO 4.4.1 guidance]*	Described here in the MDS SMP.
24.	***Phases, along with activities or milestones to be performed during each phase.*** *[same requirement as 5.1a] [ISO 4.4.2 guidance]*	Lifecycle phases described here in MDS SMP.
25.	***Phase outputs, including any documentation*** *[same requirement as 5.1b] [ISO 4.4.2 guidance]*	Included in lifecycle phase descriptions. See also Appendix C.
26.	***Verification activities (e.g., reviews, demonstrations, tests) by phase*** *[same requirement as 5.1c]*	See MDS SMP and MDS Verification documents.
27.	***o Project organization and technical interfaces: team structure; nature of project interfaces, both internal and external; roles and***	See MDS PIP.

	responsibilities, including customer responsibilities; use of subcontractors; and other crucial dependencies, such as critical equipment and facilities, and use of JPL support services. *[5.4.1c] [ISO 4.4.3, ISO 4.4.2 guidance]*	
28.	***o Project schedule.*** *[5.4.1d] [ISO 4.4.2 guidance]*	See Project WMAs.
29.	***o Risk assessment.*** *[5.4.1e] [ISO 4.4.2 guidance]*	See MDS Software Quality Assurance Plan.
30.	***Cost estimate/budget that summarizes the cost of the personnel and other resources required by the development.*** *[same requirement as 6.10.1] [ISO 4.4.2 guidance]*	See Project WMAs.
31.	***o Staffing profile.*** *[5.4.1f] [ISO 4.4.2 guidance]*	
32.	***o Change control procedures for documenting, reviewing, approving, and communicating requirements changes to all affected parties.*** *[5.4.1g] [ISO 4.3.3, 4.4.9, and 4.5]*	See MDS Information Management Plan and MDS Configuration Management Plan.
33.	***o Change control procedures for documenting, reviewing, approving, and communicating design changes before their implementation.*** *[5.4.1h] [ISO 4.4.9, 4.5]*	See MDS Information Management Plan and MDS Configuration Management Plan.
34.	***o Review (or verification) policies and procedures that, at a minimum, address detailed technical reviews, and identify what is to be reviewed (including critical intermediate products) and when reviews are to be held.*** *[same requirement as 6.9.1] [ISO 4.4.5, 4.4.6, 4.4.7]*	See MDS SMP.
35.	***o Procedures for verifying, storing, protecting, and maintaining items (e.g., software, data, hardware, specifications) supplied by the customer or designated third party.*** *[same requirement as 6.7.1] [ISO 4.7]*	See MDS Information Management Plan, MDS Configuration Management Plan and MDS Verification Plan.
36.	***o Procedures for verifying purchased or subcontracted products.*** *[same requirement as 6.6.3] [ISO 4.6.4, 4.10.2]*	See MDS Verification Plan.
37.	***o Documentation plan and procedures.*** *[same requirement as 6.2.1and 6.2.2; refer to actual requirements for elaboration] [ISO 4.5]*	See MDS SMP.
38.	***o Scope and content of the training to be provided to project personnel.*** *[same*	Covered in MDS SMP and in MDS customer delivery plan.

	requirement as 6.8.1] [ISO 4.18, 4.1.2.2]	
39.	***o System administration plan, including approach to back-up/archiving, security, and virus protection.*** *[5.4.1i] [guidance in ISO 4.4.2, 4.9, 4.15.2, 4.15.3, 4.15.5]*	Will be in configuration management section of MDS Information Management Plan
40.	***o Definition of responsibility, and description of associated procedures, to identify and correct recurring problems in the development process.*** *[5.4.1j] [ISO 4.14.1 and 4.14.3]*	Part of MDS risk management approach. See MDS Software Quality Assurance Plan.
41.	***o Metrics tailored to project needs, and the associated procedures for collecting, storing, and analyzing them.*** *[same requirement as 6.4.1]*	MDS metrics approach is documented in the MDS Software Quality Assurance Plan
42.	***o Planning of the following specific activities, including identification of any separate plans:*** *[5.4.1k]*	MDS SMP points to the other MDS process/plan documents
43.	***Configuration management*** *[same requirement as 6.1.1; refer to actual requirement for elaboration] [guidance in ISO 4.8 and 4.5]*	See MDS Configuration Management Plan and MDS Information Management Plan
44.	***Integration and test*** *[same requirement as 5.8.1; refer to actual requirement for elaboration] [ISO 4.10.1]*	Covered in MDS Verification Process Guidelines and in MDS Validation Plan
45.	***Delivery and installation*** *[same requirement as 5.9.1; refer to actual requirement for elaboration] [ISO 4.9, 4.15.1]*	Details should be covered in MDS customer delivery plan
46.	***Maintenance*** *[same requirements as 5.10.1 and 5.10.2; refer to actual requirements for elaboration] [ISO 4.19, 4.4.2 guidance, 4.4]*	The amount of maintenance that MDS provides a customer should be covered in MDS customer delivery plan.
47.	***o Reuse strategy, if any, or identification of reusable elements — both those that can be adapted from previously implemented systems, and portions of the current application that will be designed for reuse.*** *[5.4.1l]*	MDS reuse strategies are covered in MDS architecture document and in lower level design documents.
48.	***o Identification of quality records, associated procedures, and retention times.*** *[same requirements as 6.3.1 and 6.3.3; refer to actual requirements for elaboration] (See Section 6.3–Quality Records.) [ISO 4.16]*	See MDS SMP.
49.	***o Provisions for updating the plan as development proceeds.*** *[5.4.1m] [ISO 4.4.2]*	Plan will be updated as needed. Plan will be re-evaluated at start of each new customer development effort. See MDS SMP above.
50.	***o Explanations for any deviations made from SDPD requirements.*** *[5.4.1n]*	Deviations will be documented.
51.	***Requirements and design activities shall be guided by a plan with milestones and***	See MDS development schedule in MDS customer delivery plan. See MDS software lifecycle

	detailed technical reviews tailored to the needs of each project/task. *[5.6.1] [ISO 4.4.2, 4.4.6, 4.4.7]*	documented in SMP.
52.	***The design shall be documented and, prior to release, the resulting design documentation shall be reviewed to ensure that (a) the design meets the requirements and is responsive to acceptance criteria, (b) the design is verifiable, and (c) safety issues have been addressed.*** *[5.6.2][ISO 4.4.5]*	MDS SMP calls out peer reviews of designs and design documentation. MDS Software Quality Assurance personnel are also tasked with reviewing software artifacts for compliance.
53.	***Implementation activities shall be guided by one or more plans with milestones and detailed technical reviews tailored to the needs of each project/task.*** *[5.7.1] [ISO 4.4.2, 4.4.6, 4.4.7, 4.10.3]*	MDS SMP outlines MDS software lifecycle approach.
54.	***Software integration and testing shall be performed in accord with test planning and specification documentation that addresses:*** *[5.8.1] [ISO 4.4.2 guidance, 4.10.1]*	See MDS Verification Plan.
55.	***o Test requirements, which may be an elaboration of software requirements.*** *[5.8.1a] [ISO 4.10.1 guidance]*	MDS verification activity will include requirements validation and verification.
56.	***o Levels of testing required up to acceptance by the customer.*** *[5.8.1b] [ISO 4.10.1 guidance*	See MDS Validation Plan and MDS customer delivery plan.
57.	***o Test cases, test procedures, test data and expected results.*** *[5.8.1c] [ISO 4.10.1 guidance]*	See MDS Validation Plan and MDS SMP.
58.	***o Method of documenting test status and results.*** *[5.8.1d] [ISO 4.10.1 guidance, 4.12]*	See MDS Verification Process Guidelines.
59.	***o Test environment, such as dedicated processors, test tools (purchased or developed), and user documentation.*** *[5.8.1e] [ISO 4.10.1 guidance] The actual test environment should be defined precisely enough to ensure repeatability.*	Described at MDS web site.
60.	***o Approach for evaluating test tools, with respect to their ability to verify the product under test (e.g., through testing, published reviews).*** *[5.8.1f] [ISO 4.11.1]*	Described at MDS web site.
61.	***o Procedures for correcting defects, including analyzing the cause of the defect, determining corrective action, and ensuring that the corrective action is taken.*** *[5.8.1g] [ISO 4.13, 4.14.1, 4.14.2]*	See MDS SMP, MDS Software Quality Assurance Plan MDS Configuration Management Plan, and MDS Verification Plan.
62.	***Before delivery and acceptance by the customer, the developer shall validate the***	See MDS Verification Plan. MDS tests to MDS reference mission. This mission has many aspects

	product under conditions similar to the user' s application environment. *[5.8.2] [ISO 4.4.8, 4.10.4]*	that are similar to customer mission. Customer supplies mission scenarios as inputs for MDS test effort.
63.	***Missing or deficient functionality (in light of customer/user expectations, based on a requirements document or other form of "contractual" document) shall be documented in a release description document or transfer agreement.*** *[5.8.3] [ISO 4.13.2]*	MDS SMP specifies an RDD
64.	***Test records to be maintained as part of the project quality record shall include, at a minimum:*** *[5.8.4]*	See MDS SMP for list of quality (control) records.
65.	***o Anomaly reports (or problem/failure reports)*** *[5.8.4a][ISO 4.13.2]*	MDS will keep problem reports
66.	***o Test tool checks, to evaluate whether the tools are capable of verifying the acceptability of the software product under development.*** *[5.8.4b] [ISO 4.11.1]*	MDS verification team is evaluating/verifying test tools.
67.	***o Test results, with clear indications whether the product has passed or failed.*** *[5.8.4c] [ISO 4.10.5, 4.4.7]*	See MDS Verification Plan
68.	***The activities comprising delivery, installation, and acceptance shall be defined in a plan or related documentation, that addresses:*** *[5.9.1] [ISO 4.4.2 guidance, 4.9, 4.15.1]*	Defined in MDS customer delivery plan.
69.	***o Preparation of the acceptance test cases and acceptance criteria, with developer's responsibilities (if any) noted.*** *[5.9.1a] [ISO 4.10.4 guidance; based on ISO 4.4.8 and 4.10.5]*	Covered in MDS verification plan and in MDS customer delivery plan.
70.	***o Procedures to be used in documenting and resolving problems found following installation, whether during acceptance testing or delivery.*** *[5.9.1b] [ISO 4.14.1, 4.14.2]*	Covered in MDS customer delivery plan.
71.	***o Details of delivery and installation logistics, e.g., arranging for use of customer/user facilities and personnel in installation and test.*** *[5.9.1c] [ISO 4.9 guidance]*	To be worked at customer SRCR and to some extent in MDS customer delivery plan.
72.	***o Definition of developer's role (if any) in supporting transition to full operational use of the product.*** *[5.9.1d] [ISO 4.9 guidance]*	Specified in MDS customer delivery plan
73.	***o Identification of documentation to be delivered at installation, including installation and configuration procedures.*** *[5.9.1e] [ISO 4.9 guidance] Section 6.2 and Appendix B identify the documentation recommended for each class of software.*	See MDS SMP and MDS customer delivery plan.

74.	***o Identification of training for the user and/or system administrator/operator.*** *[5.9.1f] [ISO 4.9 guidance]*	See MDS PIP for discussion of training
75.	***o A schedule for key events pertaining to delivery, installation, and acceptance.*** *[5.9.1g] [ISO 4.9 guidance]*	Details to be covered in MDS customer delivery plan
76.	***o Storage of archived software media to prevent deterioration and facilitate disaster recovery.*** *[5.9.1h] [guidance in ISO 4.15.3 and 4.15.5]*	To be covered in Configuration Management section of MDS Information Management Plan
77.	***o Virus protection of software designated for delivery, during storage and electronic transmission.*** *[5.9.1i] [guidance in ISO 4.15.2, 4.15.3, 4.15.6]*	See MDS Software Quality Assurance Plan, & MDS Configuration Management Plan
78.	***After delivery, a baselined copy of the software and delivered documentation shall be archived.*** *[5.9.2]*	See MDS Configuration Management Plan
79.	***If the developer is tasked to perform maintenance, a maintenance plan shall be prepared, defining the scope of the activity and the developer' s approach.*** *[5.10.1] [ISO 4.4.2 guidance, 4.4, 4.19]*	Covered in MDS customer delivery plan and in updates and/or addendum to the plan.
80.	***If the developer is required to turn maintenance over to another organization, the development plan shall address the mechanism for transferring knowledge of the software to the maintainer.*** *[5.10.2]*	Covered in MDS SMP and in MDS customer delivery plan.
81.	***Configuration management procedures shall be documented and applied to deliverables: code, associated data files, and documentation.*** *[6.1.1] [guidance in ISO 4.8,4.5, and 4.13.1]*	See MDS Information Management Plan and MDS Configuration Management Plan.
82.	***For each development effort, the development plan shall define:*** *[6.2.1]*	
83.	***o Documents to be produced, e.g., document titles, form (web, file server, hard copy), content standards or guidelines to be followed.*** *[6.2.1a]*	Covered here in the MDS SMP and in the MDS Information Management Plan.
84.	***o Procedures (including responsibilities) for producing, reviewing, approving, and controlling documents.*** *[6.2.1b] [ISO 4.5.1]*	See this document and Appendix C.
85.	***Documentation procedures shall address:*** *[6.2.2]*	See this document and Appendix C.
86.	***o Which documents are subject to configuration management and at what point in the development cycle they are baselined.*** *[6.2.2a] [ISO 4.5.1 guidance]*	See this document and Appendix C. See also the MCDL at MDS DocuShare web site.
87.	***o Preparation of a master document list, or equivalent control mechanism, to identify document status, and preclude the use of invalid or obsolete documents.*** *[6.2.2b] [ISO 4.5.2]*	MCDL is maintained in the MDS electronic library. MDS DocuShare Library has mechanisms for always providing the most current version of a document.

88.	***o Responsibility for approving and releasing documents, and promptly withdrawing obsolete documents from use.*** *[6.2.2c] [ISO 4.5.3, 4.5.2]*	Document approvers noted on documentation covers. DocuShare library maintains current documentation version. Librarian has responsibility.
89.	***o Identification of changes in released documents (to be done where practicable).*** *[6.2.2d] [ISO 4.5.3]*	MDS official documents maintain a change page at document start.
90.	***o Approach for ensuring that the master document list (or equivalent control mechanism), as well as pertinent versions of documents, are readily available.*** *[6.2.2e] [ISO 4.5.2]*	MCDL maintained in MDS electronic library.
91.	***o Directory/file permissions and back-up policies, where document control is achieved through electronic means.*** *[6.2.2f] [ISO 4.5.2 guidance]*	Will be covered in the configuration management section of the MDS Information Management Plan
92.	***The development plan shall identify the pertinent quality records and describe procedures for collection, indexing, filing, storage, access, maintenance, and disposition of these records.*** *[6.3.1] [ISO 4.16]*	This document identifies pertinent MDS quality records. See MDS Information Management Plan for their processing.
93.	***Required quality records include the following:*** *[6.3.2]*	See this document.
94.	***o Approved changes in requirements*** *[same requirement as 5.3.7]*	MDS will use change control process as called out in MDS SMP, MDS Software Quality Assurance Plan and MDS Configuration Management Plan
95.	***o Review (or verification) results*** *[same requirements as 5.2.2–Commitment Review findings, and 6.9.2]*	MDS SMP calls out review slides as quality (control) records.
96.	***o Anomaly reports*** *[same requirement as 5.8.4a]*	MDS will have an anomaly reporting system. See this document, MDS Software Quality Assurance Plan and MDS Configuration Management Plan.
97.	***o Checks of test tools, to evaluate whether the tools are capable of verifying the acceptability of the software product under development.*** *[same requirement as 5.8.4b]*	See MDS Verification Plan.
98.	***o Test results, with clear indications whether the product has passed or failed.*** *[same requirement as 5.8.4c]*	See MDS Verification Plan.
99.	***o Change requests/orders generated during development and — if provided for in the contract — after delivery.*** *[6.3.2a] [based on ISO 4.4.9]*	See MDS SMP, MDS Software Quality Assurance Plan and MDS Configuration Management Plan.
100.	***The retention times for project/task quality records shall be established in the development plan in accord with program office directives, with particular attention to needs of post-delivery maintenance.*** *[6.3.3] [ISO 4.16]*	Covered in the MDS PIP or the MDS Information Management Plan.

101.	***Quality records shall be stored in an environment conducive to the prevention of deterioration and loss, and in a manner so as to be readily retrievable.*** *[6.3.4] [ISO 4.16]*	See MDS Information Management Plan.
102.	***Pertinent subcontractor quality records shall be identified in the subcontract.*** *[same requirement as 6.6.2b] [ISO 4.16]*	See MDS PIP for details.
103.	***Metrics, and the associated procedures for collecting, storing, and analyzing them, shall be identified in a development plan, and shall be tailored to project needs.*** *[6.4.1] [ISO 4.16]*	See MDS Software Quality Assurance Plan for details.
104.	***Purchase orders shall clearly describe the product or service ordered, and shall be reviewed for adequacy by the developer prior to release.*** *[6.6.1][ISO 4.6.3]*	See MDS PIP for details.
105.	***A development subcontract shall address:*** *[6.6.2]*	See MDS PIP for details.
106.	***o In-process verification of subcontracted development, via reviews of intermediate products and/or other oversight activities as appropriate.*** *[6.6.2a] [ISO 4.6.2b]*	See MDS PIP for details.
107.	***o Identification of subcontractor quality records to be maintained.*** *[6.6.2b] [ISO 4.16]*	See MDS PIP for details.
108.	***o Criteria and/or procedures for accepting subcontracted software.*** *[6.6.2c][ISO 4.10.2 guidance]*	See MDS PIP
109.	***Upon receipt, the developer shall ensure that a product or service that is purchased/subcontracted, or provided by a separate development organization, conforms to specified requirements, in accordance with procedures defined in the development plan.*** *[6.6.3] [ISO 4.6.1. 4.6.4, 4.10.2]*	See MDS PIP.
110.	***The developer shall establish and document procedures for verification, storage, protection, and maintenance of items (e.g., software, data, hardware, specifications) supplied by the customer or designated third party.*** *[6.7.1] [ISO 4.7]*	See MDS PIP and this document for details. MDS stores customer inputs in the MDS electronic library.
111.	***The scope and content of the training to be provided to project personnel (e.g., development team, user, maintainer) shall be addressed in the development plan.*** *[6.8.1] [ISO 4.18 and 4.1.2.2]*	The MDS PIP, MDS SMP and MDS customer delivery plan describes training to be provided both to MDS personnel and to customer personnel.
112.	***The development plan shall define review (or verification) policies and procedures that, at a minimum, address detailed technical reviews, and identify what is to be reviewed***	Addressed in MDS PIP and in the MDS SMP.

	(including critical intermediate products) and when reviews are to be held. *[6.9.1] [ISO 4.4.5, 4.4.6, 4.4.7]*	
113.	***Review (or verification) results shall be maintained as quality records, and shall include a summary of requests for action and the responses thereto.*** *[6.9.2] [ISO 4.4.6]*	Specified in MDS SMP.
114.	***For each new development, or incremental development of an existing system, the developer shall prepare a documented cost estimate/budget that summarizes the cost of the personnel and other resources required by the development.*** *[6.10.1] [ISO 4.4.2 guidance]*	See MDS SMP. Part of MDS customer delivery plan, delivery plan updates if change from baseline is negotiated

APPENDIX C: EXCEL SPREADSHEET ITEMIZING MDS PRODUCTS

				MDS Work Products	
Products	Products	Products	Products	Responsibility	Version Cntrl: What & When
MOU of Understanding	Feasibility: Management - Customer MOU	MDS Management and Systems Engineering Team negotiate product delivery with customer	Project Personnel supply details	Applicable Project Managers (or designees)	Baselined with sign off of MOU during Feasibility Phase, put in DocuShare
Updated Process Documents	Feasibility: Management - Process Definition/Update	Feasibility: Management - Process Definition/Update	Feasibility: Management - Process Definition/Update	MDS Process Engineer	Process Documents updated and put under version control by elaboration part of lifecycle for current iteration
Customer Delivery Plan (schedule, people, capabilities/requirements)	Elaboration: Management - Customer Contract	MDS Management and Systems Engineering Team negotiate product delivery with customer project.	Customer supplies requirements (textual reqs, mission scenarios, use cases), proposes need dates, hardware descriptions, mission plan, resource sharing, etc)	MDS Project manager (with help from MDS systems engineering), Customer Project Manager or designee	Baselined with sign off of customer delivery plan during feasibility
Rasmussen potential risk Item list and MDS formal database of risk items	Entire lifecycle: Risk Management	MDS Team identifies potential risk items once a month, Risk Management Team evaluates at least once a month, elevates accepted items to formal database	Team provides inputs as to possible risk items	MDS Risk Manager (MDS Deputy Project Manager)	Updated, baselined once a month
Development schedule in Microsoft Project	Entire lifecycle: Initial development in Elaboration, detailed development at start of inner loop cycle	MDS Team identifies potential risk items once a month, Risk Management Team evaluates at least once a month, elevates accepted items to formal database	Systems engineering, domain team leads negotiate phasing, more details added when phase plans developed	MDS Deputy Manager	Top level Baselined with sign off of customer delivery plan
Customer Textual Scenarios of Mission Phases	Elaboration: Requirements Analysis	MDS Team identifies potential risk items once a month, Risk Management Team evaluates at least once a month, elevates accepted items to formal database	Customer Mission Plan	Customer	Baselined with sign off of customer delivery plan
Customer Textual Requirements	Elaboration: Requirements Analysis	MDS reviews and indicates what requirements it accepts, will meet partially, and/or rejects	Customer Mission Plan	Customer	Baselined with sign off of customer delivery plan

				MDS Work Products	
Products	Products	Products	Products	Responsibility	Version Cntrl: What & When
MOU of Understanding Customer Use Cases and Sequence Diagrams	Feasibility: Management - Customer MOU Elaboration: Requirements Analysis	MDS Management and Customer Technology and Performance Use Cases/Sequence Diagrams	Project Personnel supply Customer Mission Family Mission Scenarios	Applicable Project Customer	Baselined with sign off of customer delivery plan
Rhapsody based Use Cases and Sequence Diagrams (& MDS Capabilities Catalog ?)	Elaboration: Requirements Analysis	MDS team Identifies MDS requirements as MDS Mission and System Use Cases & Sequence Diagrams	Customer requirements, X2000 system requirements and MDS architectural vision	MDS Systems Engineering Team & Domain Teams?	First baselined when Delivery Plans with customers signed updated, refined as work progresses
MDS use case requirements in DOORS	Elaboration: Requirements Analysis	Export MDS Rhapsody model to Doors	Rhapsody model	MDS Systems Engineering Team	First baselined when Delivery Plans with customers signed updated, refined as work progresses
MDS System Verification Matrix	Elaboration: Requirements Analysis	MDS Verification Team reviews top level MDS requiements, develops test matrix	MDS Rhapsody model	MDS Verification Engineer	Baseline at CDR?
MDS System Verification Plan and Procedures	Elaboration: Requirements Analysis	MDS Verification Team reviews top level MDS requirement, develops test plan and proceduues	MDS Rhapsody model	MDS Verification Engineer	Baseline at CDR?
MDS Reference Mission	Elaboration: System Analysis	Textual description of MDS reference mission	Customer mission scenarios are one source of inputs, MDS architectural vision is another	MDS Systems Engineering Team	Baselined with Customer Delivery plans
MDS Reference Scenarios	Elaboration: System Analysis	Systems engineering develops enveloping test cases for checking reference mission	Rhapsody sequence diagrams?	MDS Systems Engineering Team	First Baselined with Customer delivery plans
MDS Reference Spacecraft	Elaboration: System Analysis	Block Diagram of Reference S/C	Customer S/C Block diagrams	MDS Systems Engineering Team	First baselined with customer delivery plans
MDS Reference Ground System	Elaboration: System Analysis	Block Diagram(s) of Reference Ground System	TMOD diagrams of existing ground system from recent missions	MDS Systems Engineering Team	First baselined with customer delivery plans
Bob Rasmussen's state database	Elaboration: System Analysis	Domain experts participate in state analysis for their domain		System Architect with support from domain experts	First cut during elaboration, details during iteration

				MDS Work Products	
Products	Products	Products	Products	Responsibility	Version Cntrl: What & When
MOU of Understanding / Hardware functional block diagrams, Hardare/Software ICDs	Feasibility: Management - Customer MOU / Elaboration: System Analysis	MDS Management and Interface Trades / Hardware/Software	Project Personnel supply / StateMate Models	Applicable Project Customers? / MDS, X2000, Mission	Baselined with sign / First cut during elaboration, details during iteration
Hardware functional block diagrams, Hardare/Software ICDs	Elaboration: System Analysis	MDS Spacecraft	Customer S/C and mission scenarios, X2000 avionics block diagram	MDS Chief Engineer, MDS Systems Engineering Team	Baselined when Delivery Plans with customers signed
Textual Description	Elaboration: System Analysis	MDS Operations Concept for MDS Spacecraft/MDS mission	Operations requirements from customer, MDS vision	MDS EEIS Engineer with help from MDS system's engineering	Baselined when Delivery Plans with customers signed
Textual Description	Elaboration: System Analysis	FMECAs for X2000 hardware & low level FP	X2000 avionics hardware specifications	X2000 systems engineers	Baselined when Delivery Plans with X2000 customer signed
Rhapsody based Domain Diagram	Elaboration: Analysis - Object Analysis	Create MDS Domain Diagram with top level domain packages identified	MDS architectual vision, customer requirements	MDS Chief Engineer, MDS Chief Architect, MDS Systems Engineering, Domain leads	Baselined during elaboration during Delivery Plan creation
Rhapsody based object structural and behavioral models	Elaboration: Analysis - Object Analysis	MDS top level objects / classes are discovered/captured via sequence diagrams, class diagrams, activity diagrams	Customer and MDS requirements and system analysis products	MDS System Engineering with support from Domain leads	First baselined after team review during elaboration (peer reviews?)
Rhapsody based object & class diagrams showing collaborations (MDS system logical view), and associated text to describe analysis view	Elaboration: Analysis - Object Analysis	Class/Object collaboratios showing realization of top level use cases	Top level MDS use cases and accompanying sequence diagrams	Domain leads whose packages and classes are involved	First baselined after team review during elaboration (peer reviews?), updated as we move through inner loop cycles
Rhapsody based subsystem diagrams (deployment diagrams)	Elaboration: Architectural Design - System Architecture	Create Deployment diagrams (map large scale software components onto processors/devices)	StateMate Models that can help with software deployment trades	X2000 and MDS systems engineers	Elaboration of outer loop cycle. Presented at Peer Reviews scheduled around CDR milestone, baselined after peer reviews

				MDS Work Products	
Products	Products	Products	Products	Responsibility	Version Cntrl: What & When
Rhapsody state tables diagrams and concurrent state diagrams	Elaboration: Architectural Design - System Architecture	Create Concurrency model	StateMate models	OS / Avionics engineers with help from other domains	Elaboration of outer loop cycle. Presented at Peer Reviews scheduled around CDR milestone, baselined after peer reviews
APIs in TBD (Corba, UML, IDL ....)	Elaboration: Architectural Design - System Architecture	Create APIs for inter process communications specifications	Rhapsody class and object collaborations?	Chief engineer, systems engineering, Domain leads whose packages and classes are involved	Elaboration of outer loop cycle, refinement during inner loop cycles. Present at CDR, associated Peer Reviews
FP Patterns / ELF description	Elaboration: Architectural Design - System Architecture	Define Error Handling	Rhapsody diagrams	MDS team	Elaboration of outer loop cycle, refinement during inner loop cycles. Present at CDR, associated Peer Reviews
Tool to create Goal and Telemetry dictionaries, related software products for feeding GS	Elaboration: Architectural Design - System Architecture	Define appoach for flight ground interface for goal and telemetry processing	Source code?	Jesse Wright, etal	Present first iteration at MDS CDR?
Updated schedules as part of Phase Planning	Inner Loop: Planning/management	Fine tune contents/order of iterative proto-types, coordinate flight ground pieces	Previous schedules	John Lai with inputs from MDS team	Include in updated delivery plans, MMR presentations
APIs, textual interface agreements as needed with other MDS teams	Inner Loop: Planning/management	Each domain team develops phase plan		MDS Domain leads	Updated/expanded schedules at start of each inner loop cycle
Software Subsystem Use Cases and Scenarios for phase	Inner Loop: Analysis and Design	Subsystem Requirements analysis for current iteration	Existing use cases and scenarios in Rhapsody, refined requirements from MDS and customers	Domain lead and team members	Updated, fleshed out Rhapsody models (baseline for each iteration after design presentation). Clean up at completion of iteration

					MDS Work Products
Products	Products	Products	Products	Responsibility	Version Cntrl: What & When
MOU: Understanding	Feasibility: Management - Customer MOU	MDS Management and	Project Personnel supply	Applicable Project	Baselined with sign
Updated Rhapsody Model - Add additional classes, objects to facilitate collaborations (updated Rhapsody models)	Inner Loop: Analysis and Design	Mechanistic design for current iteration	Existing Rhapsody models	Domain lead and team members	Part of analysis and design step for current iteration, freeze at end of this step prior to implementation, update at end of iteration
Updated Rhapsody Model - Add additional classes, objects for mechanistic design (updated Rhapsody models)	Inner Loop: Analysis and Design	Identify mechanistic design patterns	Existing Rhapsody models	MDS team	Part of analysis and design step for current iteration, freeze at end of this step prior to implementation
Updated Rhapsody Model - Flesh out object guts (methods, etc) in updated Rhapsody model	Inner Loop: Analysis and Design	Detailed design for current iteration (flesh out innards of objects)	Existing Rhapsody models	Domain team member responsible for object	Part of analysis and design step for current iteration, freeze at end of this step prior to implementation
Updated System level Rhapsody models	Inner Loop: Analysis and Design / Outer Loop: Elaboration/Implementation	Refinement of design for performance (tradeoff decisions)	Existing Rhapsody models	Chief engineer, Domain leads and teams	Part of analysis and design step for current iteration, freeze at end of this step prior to implementation
Textual description - Unit Test Plan and Procedures	Inner Loop: Analysis and Design	Evaluation detailed design by development team to develop test plan / procedures	Existing Rhapsody models	Domain team software developers	Part of inner loop analysis and design and implementation
Source code (compiled)	Inner Loop: Implementation	Develop source code for objects (includes code for classes that will be inherited)	Detailed design	Team member responsible for class and/or object implementation	Freeze when ready to link into domain or be inherited by other domains
Code and textual comments - Unit tests	Inner Loop: Implementation	Domain level classes (code)	Test plan, procedures and code	Software developer in domain team	CM after tests run and code delivered for next level of integration
Textual report - Unit test result summary	Inner Loop: Implementation	write up test results	unit test results	Software developer in domain team	CM documentation (put in DocuShare?)
Compiled / linked / code within Domain	Inner Loop: Implementation	Compile and link code at domain level	Source code	Member of Domain team responsible for integration at this level	Freeze when ready to link into subsystem

				MDS Work Products	
Products	Products	Products	Products	Responsibility	Version Cntrl: What & When
MOU of Understanding Domain integration tests	Feasibility: Management - Customer MOU Inner Loop: Evaluation and Test	MDS Management and Perform first level of integration tests	Project Personnel supply Test plan, procedures and code	Applicable Project MDS I&T lead	Baselined with sign CM after tests run and code delivered for next level of integration
Domain Integration test summary	Inner Loop: Evaluation and Test	write up test results	results	MDS I&T lead	CM documentation
Compiled / linked code within one MDS node (physical subsystem)	Inner Loop: Evaluation and Test	Compile / link code for S/C or GS system physical node	Source code	MDS I&T lead	Freeze when ready to produce new MDS build
Subsystem level integration tests	Inner Loop: Evaluation and Test	Perform second level of integration tests	Test plan, procedures and code	MDS I&T lead	CM after tests run and code delivered for next level of integration
Subsystem integration test summary	Inner Loop: Evaluation and Test	write up test results	test results	MDS I&T lead	CM documentation
Compiled / linked code for one MDS physical system like S/C or entire GS	Inner Loop: Evaluation and Test	Compile and link code for MDS system	Source code	MDS I&T lead	Freeze when ready to release new MDS version
System level integration tests	Inner Loop: Evaluation and Test	Perform subsystem level tests	Test plan, procedures and code	MDS I&T lead	CM after tests run and code delivered for next level of integration
System test summary	Inner Loop: Evaluation and Test	Write up test summary	test results	MDS I&T lead	CM documentation
Compiled / linked code for entire end-to-end MDS system	Inner Loop: Evaluate and Test	Domain, subsystem and system integration and test activities	Source code	Domain teams, MDS I&T lead	Freeze for current level of testing (updates for bug fixes during test as needed)
MDS wide integration tests	Inner Loop: Evaluate and Test	Compile and link for entire system and run integration tests	Test plan, procedures and code	Domain teams, MDS I&T lead	CM after tests run and code delivered for next level of integration
End-to-end test summary	Inner Loop: Evaluation and Test	write up test results	test results	Domain teams, MDS I&T lead	CM documentation
Compiled / linked code for entire MDS system	Outer Loop: Construction	Build up entire MDS system from I&T version	Source code	MDS Verfication Team	Frozen start of testing (some bug fixes may be made during test)
MDS system verification	Outer Loop: Construction	Validate/Verify integrated MDS system against requirements	Test plan, procedures and code	MDS Verification Engineer	Code frozen for delivery/release at end of this testing

				MDS Work Products	
Products	Products	Products	Products	Responsibility	Version Cntrl: What & When
MOU of Understanding MDS test summary	Feasibility: Management - Customer MOU Outer Loop: Construction	MDS Management and write up results	Project Personnel supply test results	Applicable Project MDS Verification Engineer	Baselined with sign CM documentation
MDS software system, associated documentation, accompanying infrastructure	Outer Loop: Transition	Release current baseline MDS system to customers	MDS work products leading up to completed MDS system	MDS System Engineering	Complete system captured and archived at SRCR
Schedule and IOM Documenting Post Delivery Fixes/Update (could be Rev to Delivery Plan)	Outer Loop: Transition Management Post Delivery Maintenance	Work bug fixes / enhancements with customers		MDS Management and MDS System Engineering	Freeze level of effort with sign off of IOM

Products	Identification	Comments
MOU of Understanding	Textual Document with version #, signatures	Updates may be negotiated after original sign-off, Revisions would also require project level approval
Updated Process Documents	Textual documents with version #	Updated documents maintained within DocuShare at MDS web site. Process tweaks can occur mid cycle if necessary
Customer Delivery Plan (schedule, people, capabilities/requirements)	Textual Document with version #, signatures	Customer inputs are evaluated, accepted or rejected. Note: add fields to customer Doors database for tracking. Baseline plan by customer PDCR
Rasmussen potential risk Item list and MDS formal database of risk items	Databases with version control	Plan is to update, evaluate risk items once a month, report status at MDS MMRs
Development schedule in Microsoft Project	Schedules with date, version #?	Schedule updates covered at MMRs once monthly. At beginning of inner loop cycle a development team expands/adds details to schedule
Customer Textual Scenarios of Mission Phases	Word docs and/or excel spreadsheets	
Customer Textual Requirements	Doors Database	Expect updates prior to each new MDS build cycle

Products	Identification	Comments
MOU of Understanding Customer Use Cases and Sequence Diagrams	Textual Document Rhapsody Diagrams	Updates may be negotiated
Rhapsody based Use Cases and Sequence Diagrams (& MDS Capabilities Catalog ?)	Rhapsody Diagrams	
MDS use case requirements in DOORS	We need end-of-year Rhapsody update for Q of Service Requirements to be moved into Doors Shadow module	
MDS System Verification Matrix	Excel spreadsheet? Store in DocuShare	
MDS System Verification Plan and Procedures	Word document in CM in DocuShare	
MDS Reference Mission	Word docs placed in DocuShare	
MDS Reference Scenarios	Rhapsody Diagrams under CM	
MDS Reference Spacecraft	Diagram(s) placed in DocuShare	
MDS Reference Ground System	Diagram(s) placed in DocuShare	
Bob Rasmussen's state database	Bob Rasmussen's database	

Products	Identification	Comments
MOU of Understanding Hardware functional block diagrams, Hardare/Software ICDs	Textual Document Hardware/Software ICDs for X2000 hardware	Updates may be negotiated MDS will have several concurrent customers all using a delivery of X2000 core avionics
Hardware functional block diagrams, Hardare/Software ICDs	Diagram with Version #	
Textual Description	Textual description with version #	
Textual Description	Textual descriptions, FMECA tables, etc	Need to get cross-cutting agreement as to what FP will be implemented by MDS for customers, including X2000
Rhapsody based Domain Diagram	Diagram with Version #, includes textual mission statements	
Rhapsody based object structural and behavioral models		Elaboration of outer loop cycle, presented at Peer Reviews scheduled around PDR
Rhapsody based object & class diagrams showing collaborations (MDS system logical view), and associated text to describe analysis view	Rhapsody Diagrams under CM	Elaboration of outer loop cycle, presented at Peer Reviews scheduled around PDR
Rhapsody based subsystem diagrams (deployment diagrams)	Rhapsody Diagrams under CM	Architectural design decisions must be captured in Rhapsody in the text boxes associated with diagrams, packages, classes, etc

Products	Identification	Comments
MOU of Understanding Rhapsody [illegible] using diagrams and concurrent state diagrams	Textual Document Rhapsody Diagrams under CM	Updates may be negotiated
APIs in TBD (Corba, UML, IDL ....)	Rhapsody Diagrams under CM	
FP Patterns / ELF description		Design decision and approach need to be documented
Tool to create Goal and Telemetry dictionaries, related software products for feeding GS	IOM with Revsion#, software with Revision #	TBD tool/ approach that extracts information from flight code and feeds into GS tools, expect IOM documenting approach
Updated schedules as part of Phase Planning	After system architecture is defined we need to refine schedules	We want to implement thin vertical slices of system with risky stuff first
APIs, textual interface agreements as needed with other MDS teams		
Software Subsystem Use Cases and Scenarios for phase	Rhapsody Diagrams under CM	Bruce shows subsystem use cases being generated during object analysis, this could be a further refinement of those use cases

Products	Identification	Comments
MOU of Understanding Updated Rhapsody Model - Add additional classes, objects to facilitate collaborations (updated Rhapsody models)	Textual Document Updated, fleshed out Rhapsody models	Updates may be negotiated We add, organize classes to support particular implementation strategy. A set of classes and object collaborating together is called a mechanism...
Updated Rhapsody Model - Add additional classes, objects for mechanistic design (updated Rhapsody models)	Updated, fleshed out Rhapsody models	Reuse needs to be considered here
Updated Rhapsody Model - Flesh out object guts (methods, etc) in updated Rhapsody model	Updated, fleshed out Rhapsody models	Add textual design details within Rhapsody as way of documenting design
Updated System level Rhapsody models	Updated, fleshed out Rhapsody models	make design choices that will result in best overall system performance wise
Textual description - Unit Test Plan and Procedures	Textual documentation under CM	
Source code (compiled)	Source code under CM	There may be source code developed for base classes that will be inherited by other teams
Code and textual comments - Unit tests		
Textual report - Unit test result summary		
Compiled / linked / code within Domain		

Products	Identification	Comments
MOU of Understanding Domain integration tests	Textual Document	Updates may be negotiated
Domain Integration test summary		
Compiled / linked code within one MDS node (physical subsystem)		
Subsystem level integration tests		
Subsystem integration test summary		
Compiled / linked code for one MDS physical system like S/C or entire GS		MDS system level verification and validation takes place before release
System level integration tests		
System test summary		
Compiled / linked code for entire end-to-end MDS system		When debugging at domain and subsystem level team may be linking new software into previously baselined software
MDS wide integration tests		
End-to-end test summary		
Compiled / linked code for entire MDS system		
MDS system verification		Code should not change from end of integration testing unless bug fix essential and has project level approval for fix

Products	Identification	Comments
MOU of Understanding MDS test summary	Textual Document	Updates may be negotiated Include as part of SECR documentation
MDS software system, associated documentation, accompanying infrastructure		
Schedule and IOM Documenting Post Delivery Fixes/Update (could be Rev to Delivery Plan)		Once a final version is released to a customer per the original delivery plan these updates can be captured as revisions to the original IOM or as a new contract with the customer